

Proyecto de Sistemas Informáticos. Curso 2007-08

Aplicación web para la geolocalización y monitorización en tiempo real de los recursos integrantes de una red Grid

Autores:

Julio Alcón Ayuso
Francisco Javier Arauz Méndez
Iván Carmona Berriguete

Profesor director:

Ignacio Martín Llorente

Facultad de Informática. Universidad Complutense de Madrid



1. AUTORIZACIÓN	4
2. RESUMEN	5
2.1. Resumen en castellano.....	5
2.2. Resumen en inglés	5
3. PALABRAS CLAVE PARA BÚSQUEDAS BIBLIOGRÁFICAS	6
4. INTRODUCCIÓN.....	7
4.1. Tecnología Grid.....	7
4.1.1. Redes Grid	7
4.1.2. Globus toolkit	8
4.1.3. Metaplanificación (GridWay).....	9
4.1.4. Arquitectura del Grid:.....	11
4.1.5. Campos de aplicación.....	12
5. OBJETIVOS INICIALES	13
6. SOLUCIONES EXISTENTES	15
7. NOVEDADES Y MEJORAS QUE APORTAMOS.....	19
8. IMPLEMENTACIÓN	20
8.1. Tecnologías utilizadas	20
8.1.1. Google Maps (Introducción)	20
8.1.2. Google Maps API	21
8.1.3. HTML.....	26
8.1.4. XML	26
8.1.5. Perl.....	28
8.1.6. JavaScript	29
8.1.7. AJAX	31
8.1.8. Uso de GridWay en la aplicación.....	32
8.1.9. Relación entre las tecnologías:	34
8.2. Evolución del proyecto.....	35
8.2.1. Prototipo 1 (10 de Diciembre de 2007)	35
8.2.2. Prototipo 2 (11 de Diciembre de 2007)	36
8.2.3. Prototipo 3 (12 de Diciembre de 2007)	36
8.2.4. Prototipo 4 (17 de Diciembre de 2007)	37
8.2.5. Prototipo 5 (3 de Enero de 2008)	37
8.2.6. Prototipo 6 (15 de Enero de 2008)	38
8.2.7. Prototipo 7 (16 de Enero de 2008)	39
8.2.8. Prototipo 8 (20 de Enero de 2008)	40
8.2.9. Prototipo 9 (27 de Febrero de 2008).....	41
8.2.10. Prototipo 10 (3 de Marzo de 2008).....	42
8.2.11. Prototipo 11 (2 de Abril de 2008).....	43
8.2.12. Prototipo 12 (13 de Abril de 2008).....	44
8.2.13. Prototipo 13 (19 de Abril de 2008).....	45
8.2.14. Prototipo 14 (20 de Abril de 2008).....	46
8.2.15. Prototipo 15 (27 de Abril de 2008).....	47
8.2.16. Prototipo 16 (15 de Mayo de 2008).....	48
9. DESCRIPCIÓN DE LA INTERFAZ.....	49
9.1. Recuadro de configuración.....	49

9.2. Ventana de información cartográfica	50
9.3. Mapa de visualización	51
9.4. Recuadro de selección de sitios	51
9.5. Gráfica de monitorización	52
10. SIMULACIÓN DEL FUNCIONAMIENTO DE LA APLICACIÓN:	53
10.1. Simulación a través de trazas	54
11. INSTALACIÓN Y EJECUCIÓN DE LA APLICACIÓN.....	53
11.1. Instalación de la aplicación (únicamente si se ejecuta con las trazas).....	55
11.2. Ejecución	55
12. POSIBLES AMPLIACIONES Y MEJORAS	56
13. AGRADECIMIENTOS	57
14. BIBLIOGRAFÍA Y REFERENCIAS	58

1. AUTORIZACIÓN

Por la presente, los autores de este proyecto autorizamos a la Universidad Complutense de Madrid a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores tanto de la propia memoria como el código, la documentación y/o el prototipo desarrollado.

Fdo. Julio Alcón Ayuso

Fdo. Francisco Javier Arauz Méndez

Fdo. Iván Carmona Berriguete

En Madrid, a 4 de Julio de 2008.

2. RESUMEN

2.1. Resumen en castellano

El objetivo de este proyecto es proporcionar una herramienta accesible a través de Internet que permite la monitorización en tiempo real de los recursos integrantes de una red Grid, así como mostrar su localización física. Para la tarea de monitorización se utilizan los comandos “gwhoost” y “gwps” de GridWay invocados a través de un demonio en lenguaje Perl, mientras que para la geolocalización se recurre a la conocida herramienta Google Maps.

2.2. Resumen en inglés

The goal of this project is to provide an accessible through the Internet tool which allows real-time monitorization of the integral resources from a Grid network, as well as showing its physical location. Thus, GridWay commands “gwhoost” and “gwps” are invoked by a written in Perl language demon for monitorization task, whereas well-known tool Google Maps is utilized for geolocalization.

3. PALABRAS CLAVE PARA BÚSQUEDAS BIBLIOGRÁFICAS

A continuación se muestran los términos considerados como palabras clave ordenados alfabéticamente:

- Aplicación web
- Geolocalización
- Google Maps
- Grid
- GridWay
- HTML
- JavaScript
- Monitorización
- Perl
- XML

4. INTRODUCCIÓN

Una de las principales motivaciones que ha contribuido al desarrollo de la Informática es el deseo y/o necesidad del ser humano de resolver problemas complejos. De hecho, en el siglo XVII, hombres como Blaise Pascal o Gottfried Leibniz fueron capaces de crear maquinas mecánicas de cálculo (precursoras de las modernas calculadoras y de los ordenadores) para ayudar al hombre en tareas tan dispares como la astronomía o la recaudación de impuestos, que sólo tienen en común el hecho de requerir la realización intensiva de operaciones matemáticas más o menos complejas. Sin embargo, la complejidad de los problemas a tratar se ha incrementado exponencialmente con el paso del tiempo, de forma que hoy en día se requiere una gran potencia de cálculo en campos como la meteorología o la física cuántica. Una alternativa para lograr dicha potencia de cálculo es el uso de supercomputadores, pero no obstante a veces la complejidad de los problemas a desarrollar resulta demasiado grande incluso para estos computadores, y, además, el coste de los supercomputadores resulta muy elevado, lo que hace que su uso esté limitado a organismos gubernamentales, militares y grandes centros de investigación; todo esto hace que surja como alternativa la tecnología Grid.

4.1. Tecnología Grid

4.1.1. Redes Grid

Un **grid** es un sistema distribuido, no sujeto a control centralizado, basado en interfaces y protocolos de propósito general y abierto. Este sistema es capaz de satisfacer determinados niveles de calidad de servicio en términos de seguridad, productividad del sistema, o el uso coordinado de diferentes tipos de recursos.

El proyecto usa esta tecnología para ejecutar los comandos *gwhost* y *gwps* de **GridWay** y así poder realizar la monitorización de los recursos de la red grid, mostrando su localización geográfica. La red grid recibe los comandos y devuelve el resultado de la ejecución de los mismos siendo estos tratados por la aplicación.

El hecho de que puedan estar multitud de computadores conectados en red usando grid, supone una potencia ilimitada, ofreciendo una perfecta integración de sistemas y dispositivos distintos, por lo que las conexiones entre las diferentes máquinas no generarán ningún problema.

Podemos afirmar por tanto, que la red grid, supone una solución altamente escalable, potente y flexible, ya que evitará problemas de falta de recursos y nunca quedará obsoleta, dada la posibilidad de aumentar el número de sus componentes y las características de los mismos.

Estos recursos se distribuyen en la red de forma transparente pero guardando unas pautas de seguridad y políticas de gestión de carácter tanto técnico como económico. Así pues, su objetivo será el de compartir una serie de recursos en la red de manera uniforme, segura, transparente, eficiente y fiable, ofreciendo un único punto de acceso a un conjunto de recursos distribuidos geográficamente en diferentes dominios de

administración. Esto nos puede llevar a pensar que la computación Grid permite la creación de organizaciones virtuales.

4.1.2. Globus toolkit

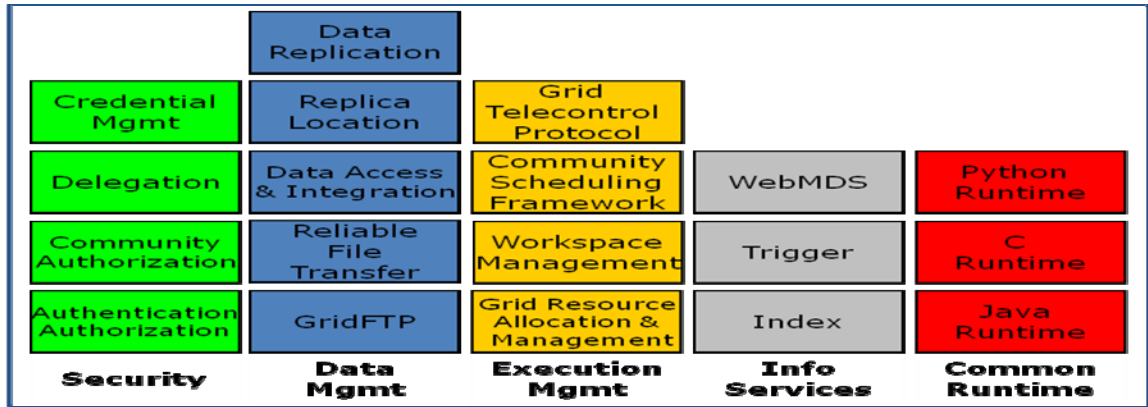


Figura: Globus Toolkit como una infraestructura Open-Source.

El código abierto de Globus Toolkit es la tecnología fundamental para la “Grid”, permitiendo compartir el poder de computación, bases de datos y otras herramientas de seguridad en línea a través de límites corporativos, institucionales y geográficos sin sacrificar la autonomía local.

Globus Toolkit incluye servicios y librerías que resultan necesarios para diversas tareas a realizar con los recursos, como pueden ser la búsqueda, gestión y monitorización de los mismos. Además incluye software para garantizar la seguridad, la infraestructura de información, el manejo de los recursos, la comunicación, detección de avería y portabilidad. Está empaquetado como un conjunto de componentes que pueden ser usados tanto de forma independiente o conjuntamente con otros entornos de desarrollo de aplicaciones con el fin de potenciar las creaciones obtenidas con estos últimos.

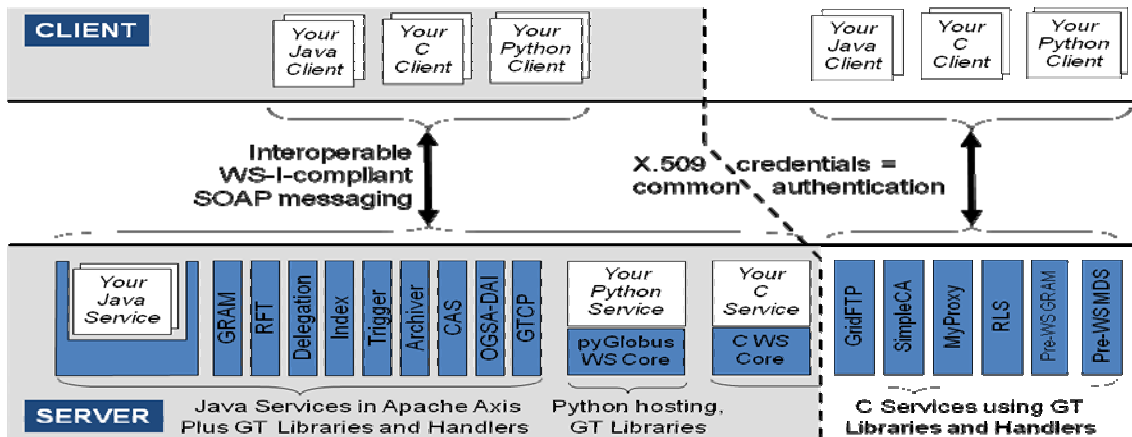


Figura: Conjunto de componentes de Globus Toolkit

Cada organización tiene sus propios modos de trabajo, y las colaboraciones entre distintas organizaciones se ven en muchos casos obstaculizadas por algo tan común como puede ser la incompatibilidad entre los archivos que quieren compartir, las distintas computadoras usadas por cada organización así como sus distintas redes de conexión.

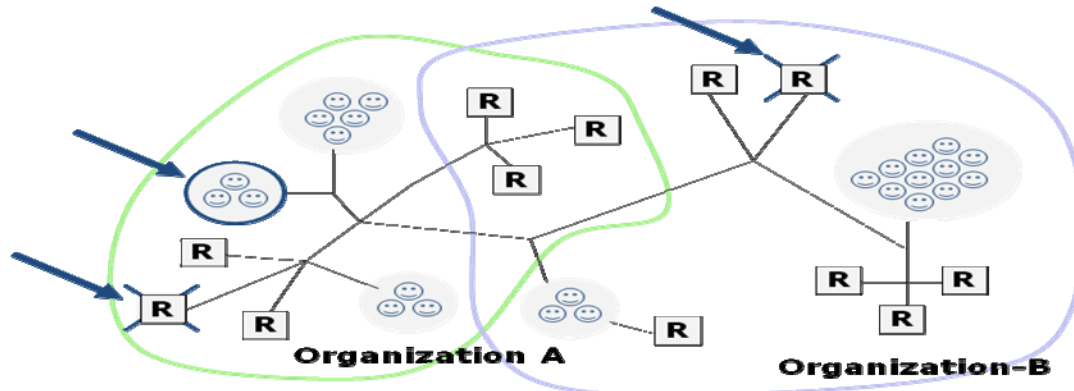


Figura: Globus permite la compartición segura de recursos entre organizaciones con filosofías de trabajo distintas.

Globus Toolkit fue concebido para eliminar todos estos obstáculos que dificultan la colaboración. Proporciona al usuario que utiliza los servicios ofrecidos, una perspectiva sobre el ámbito de trabajo remoto similar a la que obtendría realizando un acceso local a su equipo, manteniendo siempre un cierto nivel de privilegio sobre sus recursos. Esto se consigue gracias a los protocolos, interfaces y servicios del núcleo que se ofrecen.

Globus se trata de un código “Open-source”. Similar a lo que se sucede con el sistema operativo Linux, le permite estar actualizado y ser capaz de utilizar las últimas tecnologías que aparecen, gracias a la colaboración de toda la comunidad de usuarios.

Este proyecto utiliza un grid montado con Globus, a través del metaplanificador GridWay.

4.1.3. Metaplanificación (GridWay)

Gridway es un metaplanificador que permite una compartición de recursos computacionales (clusters, servidores, supercomputadoras...) de manera segura, eficiente y a larga escala, gestionado por diferentes sistemas LRM (Local Resource Management), como PBS, SGE, LSF, Condor..., con una organización simple o dispersada a través de varios dominios administrativos.

Gridway es un proyecto Globus, solapado a la filosofía Globus y con las directrices para el desarrollo colaborativo y la incorporación de contribuciones tanto de individuales como de corporaciones por todo el mundo.

Existe un número de sistemas de gestión y planificación de la carga de trabajo tanto comerciales como de libre distribución disponibles hoy en día, cada uno adecuado para diferentes infraestructuras de computadores subyacentes y perfiles de ejecución.

Gridway destaca entre otros sistemas metaplanificadores porque ha sido diseñado específicamente para trabajar por encima de los servicios de Globus, ofreciendo la más alta funcionalidad, calidad de servicio y confiabilidad en esta clase de infraestructuras.

Cuenta con muchas ventajas para:

Directores de infraestructuras y proyectos: Gridway es un proyecto comunitario de libre distribución, que cuenta con la filosofía Globus y las directrices para el desarrollo colaborativo y no necesita de nuevos servicios, sino que trabaja con los mismos que Globus.

Integradores de sistemas: Gridway es altamente modular, permitiendo la adaptación a diferentes infraestructuras grid y soporta varios estándares OGF. También es fácilmente extensible y podría ser utilizado o extendido por arquitecturas más complejas que implementen los acuerdos a nivel de servicio.

Jefes de sistemas: Gridway proporciona un marco de trabajo de planificación similar al encontrado en los sistemas de gestión de recursos locales (LRM), permitiendo llevar la cuenta de los recursos y definiendo políticas de planificación.

Desarrolladores de aplicaciones: Gridway implementa el API estándar DRMAA (C y Java), asegurando la compatibilidad de aplicaciones con sistemas LRM que implementan el estándar como SGE, Condor, Torque,...

Usuarios finales: Gridway permite a los usuarios finales enviar, monitorizar, sincronizar y controlar trabajos de forma totalmente transparente y similar a los sistemas de gestión de recursos locales, que podrían ser descritos usando el estándar OGF JSDL. El planificador les facilita el trabajo ya que puede gestionar diferentes situaciones de fallo de forma automática.

Con Gridway, una infraestructura Grid puede ser explotada y gestionada de la misma manera que un cluster local.

4.1.4. Arquitectura del Grid:

Su **arquitectura** se detalla en la siguiente figura:

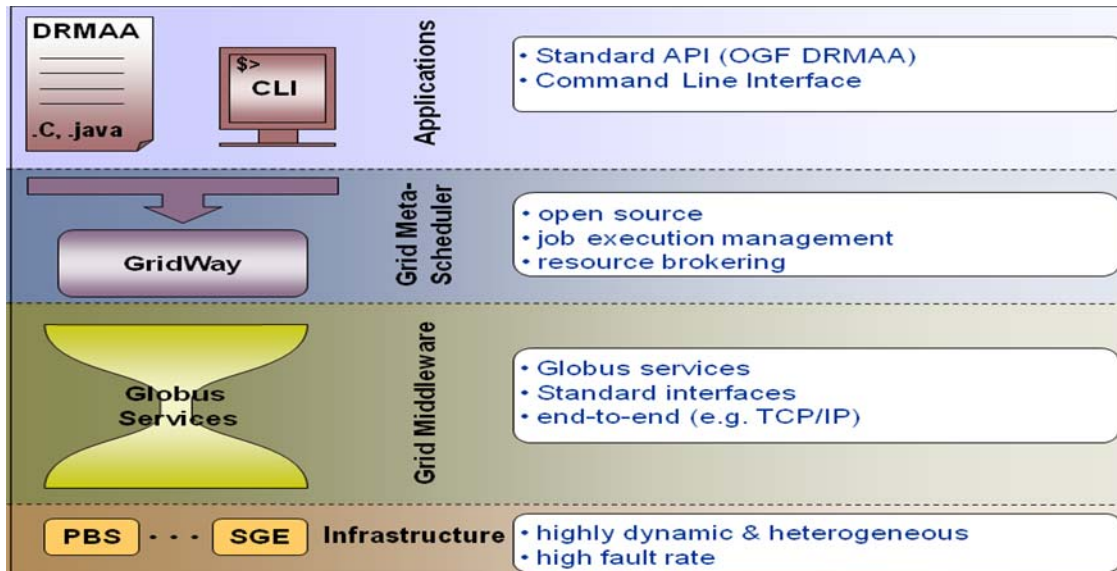


Figura: Arquitectura global de un grid computacional

La arquitectura propuesta es una arquitectura de protocolos que definen los mecanismos básicos que permiten a los usuarios y a los recursos negociar, establecer, gestionar y explotar la compartición de recursos. Una arquitectura abierta basada en un estándar, facilita la extensibilidad, la interoperabilidad, la portabilidad y la compartición de código.

En el **nivel de infraestructura** se encuentran los recursos computacionales, como son los ordenadores, los clusters, los supercomputadores, los sistemas de almacenamiento en red, las bases de datos, etc. También se incluyen en este nivel la infraestructura de la red y sus mecanismos de gestión y control.

El **nivel de conectividad** incluye los protocolos de comunicación y seguridad que permiten a los recursos computacionales comunicarse. Entre estos protocolos se encuentran: la pila de protocolos TCP/IP, el protocolo SSL, Certificados X.509. También se incluyen los nuevos protocolos que se encuentran en fase de estudio y que permitirán mejorar el rendimiento en las redes de alta velocidad.

El **nivel de recursos** engloba todos los servicios que permiten gestionar un conjunto de recursos. Se encuentran los servicios de directorio, que permiten localizar los recursos que son de nuestro interés; los schedulers distribuidos, que permiten asignar las tareas a cada recurso; la monitorización y diagnóstico de la ejecución de las distintas tareas en que se distribuyen la ejecución de una aplicación; la contabilidad, que permite calcular el coste de la utilización de varios recursos heterogéneos, y, el acceso a datos distribuidos, que gestiona la replicación de datos.

El último **nivel**, el **de aplicación**, se centra en la definición de protocolos que permitan a las aplicaciones el acceso a la infraestructura del grid a través de las distintas capas.

Según el tipo de aplicación puede ser necesario conectarse a las distintas capas o acceder directamente a una de ellas, incluso directamente a la infraestructura.

Es en esta última capa en la que se ubica la aplicación desarrollada en este proyecto.

4.1.5. Campos de aplicación

Para terminar de definir el concepto, se desgana cada uno de los posibles campos de aplicación de la tecnología grid.

Existen cinco grandes áreas de trabajo determinadas por las necesidades de cálculo, espacio para el almacenamiento de los datos y tiempo de respuesta. Las áreas son:

Supercomputación distribuida: Engloba aplicaciones cuyas necesidades no se pueden satisfacer con un sólo nodo. Estas necesidades son puntuales e intensivas. Ejemplo de este tipo de aplicaciones son las simulaciones, las herramientas de cálculo numérico, los procesos de análisis de datos, etc.

Sistemas distribuidos en tiempo real: Aplicaciones que generan un flujo de datos a alta velocidad que debe ser analizado y procesado en tiempo real. Ejemplo de este tipo de aplicaciones son los experimentos de física de alta energía, control remoto de equipos médicos de alta precisión.

Proceso intensivo de datos: Aplicaciones que hacen un uso intensivo del espacio de almacenamiento. Las necesidades de almacenamiento sobrepasan la capacidad de almacenamiento de un único nodo y los datos son distribuidos por todo el grid. Además de los beneficios por el incremento de espacio, la distribución de los datos a lo largo del grid permite el acceso a los mismos de forma distribuida. Ejemplos de este tipo de aplicaciones son todos los sistemas gestores de bases de datos distribuidas.

Servicios puntuales: En este área, nos olvidamos del concepto de potencia de cálculo y capacidad de almacenamiento para centrarnos en recursos que una organización puede considerar como no necesarios. De esta manera el grid ofrece a la organización esos recursos sin que la organización deba desarrollarlos por sí misma. Ejemplos de este tipo de aplicaciones, son aquellas que permiten acceder a hardware muy específico (equipos costosos de medida o de análisis de muestras) para la realización de labores a distancia.

Entornos virtuales de colaboración: Este área está relacionada directamente con el concepto de teleinmersión, de manera que se utilizan los enormes recursos computacionales del grid y su naturaleza distribuida para generar entornos virtuales 3D distribuidos.

Como **resumen** a lo visto en esta introducción, podemos decir que, la tecnología Grid consiste en el uso de un servicio para permitir trabajar en paralelo a recursos diferentes respetando la política de seguridad y de gestión de cada uno de ellos. Con este sistema, se consigue el acceso a más recursos con el consiguiente acceso a más memoria y velocidad de procesamiento que a los que se accedería mediante la utilización de un supercomputador, a la vez que se minimizan enormemente los costes; dichas ventajas

convierten a la computación Grid en un campo en alza dentro de la investigación informática.

Cabe destacar el hecho de que el auge de la tecnología Grid supone un desafío para la comunidad informática en cuanto a que requiere de nuevas herramientas que faciliten la comunicación entre los distintos recursos que forman parte de una determinada red Grid. Así, en ocasiones puede ser necesario acceder a informaciones tales como el estado de cada uno de esos recursos o la localización física de éstos. Y en respuesta a estas necesidades surgen aplicaciones como la que nosotros hemos implementado.

Nuestro proyecto consiste en la implementación de una aplicación accesible vía web que permite tanto monitorizar en tiempo real la actividad de los recursos integrantes de una red Grid (concretamente de una red integrada dentro del metaplanificador GridWay) como mostrar la localización física de éstos. Dicha aplicación está implementada principalmente en HTML y JavaScript, aunque también incluye elementos en XML o Perl; además, para la visualización de la localización física hemos utilizado el sistema de visualización de mapas Google Maps, puesto que dicha herramienta consta de una API que permite la integración de sus mapas en cualquier aplicación web con unos resultados altamente satisfactorios.

Para la realización de las dos tareas descritas anteriormente (monitorización y geolocalización de los recursos), en nuestro proyecto interactúan dos elementos. El primero de ellos es un demonio que se encarga de la adquisición de la información, que tras unas transformaciones, almacena en forma de archivos XML los datos obtenidos como resultado de la ejecución de los comandos de GridWay “gwhoost” y “gwps”. Por su parte, el otro elemento de nuestra aplicación es una página web que permite la visualización de los datos anteriores a través de dos vías: un mapa de Google Maps convenientemente personalizado para incluir marcadores indicando la localización de cada recurso y pestañas desplegables con información de diversa índole, y un diagrama de barras; además, dicha página web incluye diversas opciones de configuración.

A lo largo de las siguientes páginas trataremos de mostrar una visión global de lo que ha sido el desarrollo de este proyecto. Para ello, comenzaremos trazando una comparativa entre nuestra aplicación y otras herramientas similares ya existentes; después, trataremos diversos aspectos acerca de la implementación de nuestro proyecto, centrándonos principalmente en las características de las distintas tecnologías y herramientas utilizadas y en los estados por los que fue pasando nuestra aplicación hasta llegar a su situación actual; y finalmente describiremos como utilizar esta herramienta de forma que pueda aprovecharse todo el potencial que ofrece. Además, todas estas secciones se verán acompañadas de un gran número de imágenes que complementen la información explicada en cada apartado.

5. OBJETIVOS INICIALES

Como se ha mencionado anteriormente, el objetivo de este proyecto es proporcionar una herramienta accesible a través de Internet que permite la monitorización en tiempo real de los recursos integrantes de una red Grid, así como mostrar su localización física. Es más, el mapa en el que se mostraría la posición de los hosts debía ser también el elemento de representación principal y de acceso para todas las medidas de monitorización obtenidas para cada uno de los mismos.

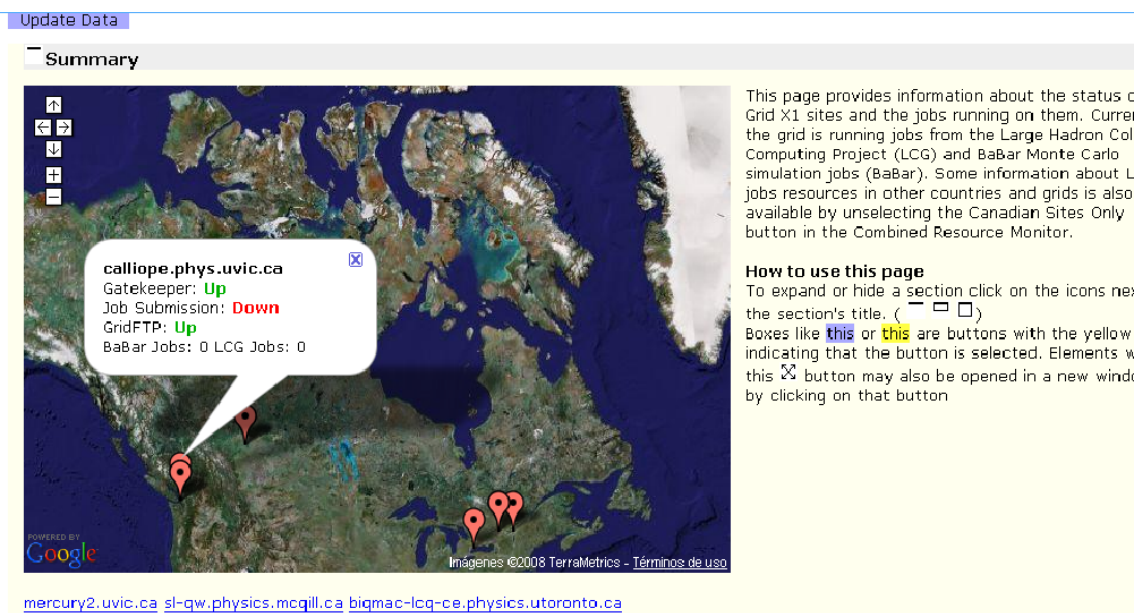
En resumidas cuentas, las características que se deseaba que tuviese la aplicación son las siguientes:

- Aplicación web, ejecutable desde cualquier navegador.
- Actualización de los datos en tiempo real.
- Posibilidad de modificar el periodo de refresco de los datos.
- Capacidad de mostrar varias métricas distintas en el mapa a elección del usuario.
- Capacidad de seleccionar los hosts mostrados en el mapa.
- Capacidad de comparar los datos estadísticos de los distintos sitios o agrupaciones del grid mediante una grafica dentro del mapa.
- Poder acceder con más detalle a toda la información disponible de cada sitio.
- Representar mediante diagramas comparativos los resultados obtenidos en la monitorización de los hosts representados en el mapa.
- Representación visual/grafica de los datos.

6. SOLUCIONES EXISTENTES

Actualmente en la red podemos encontrar algunas aplicaciones que tienen un propósito similar o parecido al de nuestra herramienta. Algunos de estos productos son también ejecutables desde el navegador Web, mientras que otros requieren una instalación previa en el equipo del usuario. Uno de los objetivos principales del proyecto era que fuese accesible vía Internet para cualquier usuario, por lo que nos centraremos solamente en las soluciones existentes online mas representativas.

En la web canadiense de “GridX1” (<http://monitor.gridx1.ca/>) encontramos un completo monitor online para un Grid. Como se puede apreciar en la imagen, también cuenta con un mapa de googlemaps.



El problema es que dicho mapa tiene escasa funcionalidad, ya que permite mostrar de forma gráfica muy poca información de toda la que ofrece la pagina. De hecho, esta se refleja en el mapa de forma estática. No se refresca de forma automática ni se da la opción de visualizar otro tipo de estadísticas. Los datos se muestran debajo del mapa desplegando una serie de opciones.

Como hemos comentado antes, el refresco de los datos no es en tiempo real sino que se realiza a través del botón “update data” situado en la parte superior izquierda de la pantalla.

Esta opción de monitorización es una buena aproximación al objetivo de nuestro proyecto, pero la información mostrada en el mapa es escasa y no se ofrece al usuario la posibilidad de distintas formas de visualización de la misma.

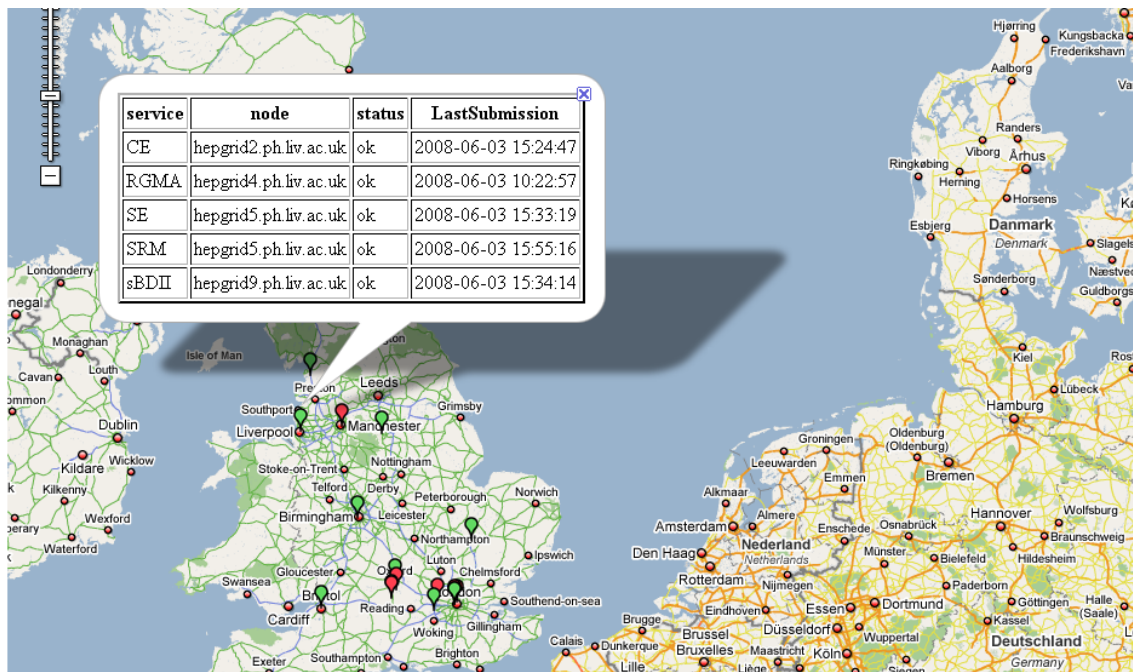
Encontramos otras soluciones en la web de GridPP (<http://www.gridpp.ac.uk/>), perteneciente a un grupo de informáticos y físicos del Reino Unido. Aquí, encontramos otro monitor que utiliza como medio de disponer la información un mapa de googlemaps así como un par de herramientas de monitorización y geolocalización.

Aplicación web para la geolocalización y monitorización en tiempo real de los recursos integrantes de una red Grid

En primer lugar disponen de un monitor en 3D para cuya utilización hay que descargar un programa y proceder a la instalación de este en el ordenador del usuario.

También tienen un monitor en 2D online. Sin embargo es complicado que este funcione ya que requiere modificar algunos recursos del sistema operativo del usuario, para que la maquina virtual de java pueda ejecutar el monitor en el navegador.

Esta aplicación ofrece también una versión de otro monitor que utiliza googlemaps <http://goc02.grid-support.ac.uk/googlemaps/gridpp.html>



Este monitor está presente en varias paginas de la red y se puede acceder a información sobre el mediante el siguiente link <http://goc.grid.sinica.edu.tw/gocwiki/GoogleMap>.

Sólo muestra la información que se ve en el dibujo, es decir el servicio que ofrece el nodo, el nombre del nodo, el estado y la última fecha de actualización de dichos datos. Los datos se refrescan cada hora.

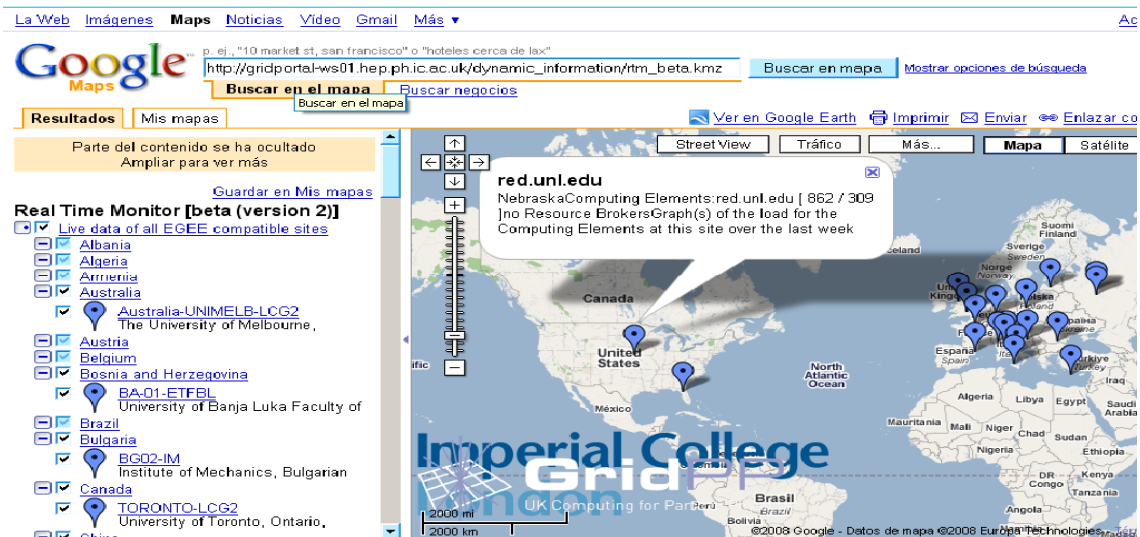
Los marcadores que indican donde se encuentran los nodos del grid están coloreados según el sitio haya pasado o no con éxito el test SAM.

Este monitor debe estar aún en fase de desarrollo y se comporta de manera diferente en Firefox y Explorer. En el se aprecian errores javascript y, a veces, si hay muchos nodos a representar dentro de cada sitio, se descuadra la información que aparece dentro de la ventana de información del mapa.

Una vez más, la principal carencia de esta solución es que los datos representados en el mapa son escasos e invariables y no se puede elegir la información a visualizar.

En esta web también se nos ofrece la opción de utilizar otro monitor perteneciente al London Imperial College utilizando el siguiente link:

http://maps.google.com/maps?q=http:%2F%2Fgridportal-ws01.hep.ph.ic.ac.uk%2Fdynamic_information%2Frtm_beta.kmz



Este monitor sólo muestra los puntos geográficos e información referente a cada nodo. El problema es que la información que presenta varía de un nodo a otro y no se puede seleccionar la información mostrada. Las ventajas que ofrece son que los datos mostrados son en tiempo real y que también dispone de una lista con los hosts que pertenecen al grid.

Esta es otra implementación de un monitor en un mapa totalmente distinta. Este mapa sí permite variar la información que se muestra, y da una idea sobre cual es el volumen de trabajo. No es geográfico pero es muy interactivo y tiene múltiples opciones de monitorización.

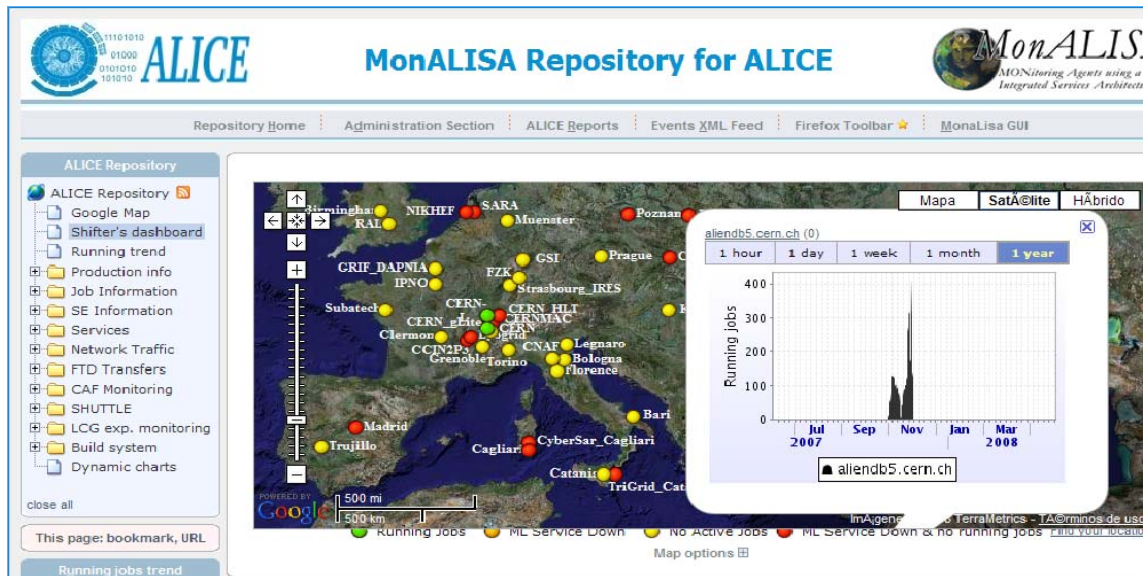
<http://gridmap.cern.ch/gm/>

GridMap Prototype – Visualizing the "State" of the Grid



Aplicación web para la geolocalización y monitorización en tiempo real de los recursos integrantes de una red Grid

Por ultimo en la pagina de ALICE GRID MonALISA Repository for ALICE (<http://pcalimonitor.cern.ch/map.jsp>) encontramos este monitor sobre googlemaps .



Como se puede apreciar en la imagen se diferencian con colores los hosts que tienen jobs activos, ejecutándose, los que están pendientes....

En la grafica se presentan estadísticas sobre el número de jobs realizados en determinados periodos de tiempo (hora, día, semana, mes, año).

En la barra de menús de la izquierda se ofrecen al usuario más posibilidades de monitorización pero ninguna de estas aparece reflejada en el mapa, si no que se cambia la pagina mostrada en el navegador.

Esta implementación es más completa y aporta más información en tiempo real al usuario, aunque tampoco se pueden elegir las métricas mostradas en el mapa.

7. NOVEDADES Y MEJORAS QUE APORTAMOS

Como se ha visto, en la mayoría de las soluciones existentes, los datos mostrados en el mapa suelen ser estáticos, y en el caso de que estos se refresquen o sean en tiempo real, no se suele poder variar la métrica o la estadística que se muestra en el mapa.

En nuestra implementación, la actualización de los datos es en tiempo real y además el usuario tiene la opción de decidir cual es el periodo de refresco de los datos representados. Además tiene total libertad de elegir las estadísticas que se muestran tanto en el mapa como en la grafica, siempre ajustándonos a los datos que devuelven los comandos gwps y gwhost.

En cuanto al acceso a los datos representados, en algunas de las soluciones existentes, solo se muestra en emplazamiento del nodo de la red grid, y para poder ver cualquiera de sus estadísticas, o simplemente el estado es necesario perder de vista el mapa.

En cambio, con nuestra solución de un solo vistazo, podemos ver la estadística seleccionada para todos los hosts que aparecen en el mapa, así como el porcentaje y la medida real de la misma. También se puede seleccionar qué hosts se visualizan y cuales no. Además en la grafica que aparece en la parte inferior de la página se pueden comparar rápidamente todos los hosts seleccionados.

Este era uno de los principales objetivos del proyecto; poder seleccionar tanto las métricas y estadísticas como los nodos de la red mostrados.

En la mayoría de los monitores existentes en la red que hemos examinado, o bien la información es muy escasa, limitándose en algunos casos a indicar el estado de la maquina y su posición, o bien es tal la cantidad de datos aportados que se hace necesario abrir una nueva pagina en el navegador para acceder a los mismos.

Gracias a la inclusión de varias pestañas dentro de la ventana de googlemaps, podemos mostrar varios niveles y agrupaciones de información para un mismo host sin tener que variar la vista. Y con cerrar esa ventana informativa volvemos a tener la vista general del mapa con la métrica seleccionada.

La herramienta es accesible desde los principales navegadores, si necesidad de ningún tipo de instalación previa, ni de modificar ningún parámetro del sistema operativo. De este modo se puede monitorizar el grid desde cualquier equipo que disponga de conexión a Internet.

Por ultimo, debido a que el propósito es que el usuario pueda de un vistazo evaluar el estado de la red Grid o de uno de sus nodos, se ha dotado de gran importancia al factor visual de la aplicación y a la sencillez de su utilización. De este modo en nuestra solución prima la nitidez en la representación de los datos y el fácil acceso a los mismos, sobre la cantidad de los resultados mostrados. Y gracias a la posibilidad de ampliar y reducir el mapa, variar las métricas mostradas y seleccionar el refresco y los hosts representados y demás opciones se consigue el efecto deseado, que el usuario pueda con velocidad, eficacia y comodidad conocer el estado de su red Grid

8. IMPLEMENTACIÓN

8.1. Tecnologías utilizadas

En este punto se hará una descripción de las herramientas utilizadas en la realización de esta aplicación. Estas se expondrán siguiendo el orden en el cual nos hemos ido familiarizando con cada una de ellas.

8.1.1. Google Maps (Introducción)

Google Maps, durante una época llamado *Google Local*, es un servidor de aplicaciones de mapas en web desarrollado con la tecnología de Google.

En sus inicios, ofrecía mapas de ciudades de diversos países (EEUU, Canadá, Reino Unido, Japón), en principio como complemento y ayuda al usuario que realiza búsquedas en *Google Local*.

Google Maps fue anunciado por primera vez en el *Google Blog* el **8 de Febrero de 2005**. Originariamente era sólo soportado por los navegadores *Internet Explorer* y *Mozilla Firefox*. El soporte para *Opera* y *Safari* fue agregado el 25 de Febrero de ese mismo año. El software estuvo en su fase beta durante 6 meses antes de convertirse en parte de *Google Local*, el 6 de Octubre de 2005

En Abril de 2005 comenzó a ofrecer además, imágenes vía satélite provenientes de la firma *Keyhole*, adquirida por Google. De esta manera, podemos ver fotografías aéreas de todo el planeta de mayor o menor resolución dependiendo si se trata o no de importantes núcleos urbanos.

Todas las imágenes de satélite de GoogleMaps son las mismas que las que podemos encontrar en otra famosa herramienta de Google llamada *GoogleEarth*.

Del mismo modo que otras aplicaciones web desarrolladas por Google, para implementar Google Maps, se usan un gran número de ficheros *JavaScript*. Como el usuario puede mover el mapa, la visualización del mismo se descarga desde el servidor. Cuando este busca un punto determinado, la ubicación está marcada por un indicador en forma de pin, el cual es una imagen *PNG* transparente sobre el mapa.

Para conseguir la conectividad sin sincronía con el servidor, y así proporcionar al usuario mayor interactividad con el mapa, mediante la realización de peticiones asíncronas a la red con *JavaScript* y *XMLHttpRequest*, se usa una técnica reciente, conocida como *AJAX*, sobre la que ofreceremos información mas adelante.

Además, en **Junio de 2005** se presentó la *API de Google Maps* que permite a los programadores crear sus propias aplicaciones web utilizando las imágenes y mapas de este servicio.

8.1.2. Google Maps API

Google Maps API se trata de una tecnología que permite la visualización de Google Maps en tus propias páginas web con JavaScript. El API proporciona unas determinadas herramientas para interaccionar con los mapas y añadir contenido a los mismos a través de una serie de servicios, permitiendo llegar a crear aplicaciones con mapas de gran complejidad y robustez.

Para implantar esta tecnología en tu página web, google te proporciona una clave “single Maps API key”, que es válida para un único directorio o dominio. Para obtener esta clave se debe tener una cuenta de google, y la clave que se proporciona estará conectada a dicha cuenta.

La obtención de esta clave sólo nos ha hecho falta para la instalación definitiva de la aplicación en la máquina “draco” de la Universidad Complutense de Madrid. Ya que, para hacer nuestras pruebas y para el desarrollo posterior, hemos trabajado sobre htmls que podían ser testados sin la necesidad de incluirlos en un dominio o directorio determinado.

La documentación que nos encontramos, estaba diseñada para gente familiarizada en programación con JavaScript y con conceptos de programación orientada a objetos. Así que a la vez que experimentábamos con las herramientas que nos proporciona google íbamos familiarizándonos con la programación en JavaScript a través de los muchos tutoriales que se pueden encontrar en la red.

Google ofrece dos tipos de documentación: Una de ellas, está diseñada para permitir al nuevo usuario, empezar rápidamente a experimentar y desarrollar sencillas aplicaciones con Google Maps API. El otro tipo de documentación que se ofrece, se trata de una guía de referencia completa y exhaustiva: Google Maps API Reference . Recientemente esta documentación se ha reorganizado para ofrecer una información más conceptual y así focalizar las posibles discusiones en las distintas áreas dominantes.

En esta pequeña introducción a modo de guía para los nuevos usuarios lo primero que se explica es cómo mostrar el mapa en tu página y las operaciones básicas sobre el mismo. Para ello, seguimos los pasos siguientes:

1 .Cargar el Google Maps API

```
<script src="http://maps.google.com/maps?file=api&v=2&key=abcdefg"
    type="text/javascript">
</script>
```

La URL *http://maps.google.com/maps?file=api&v=2&key=abcdefg* apunta al fichero JavaScript que incluye todos los símbolos y definiciones necesarias para usar el Google Maps API.

2. Map DOM Elements

```
<div id="map_canvas" style="width: 500px; height: 300px"></div>
```

Para poder mostrar el mapa en la web, debemos reservar un hueco para él. Esto lo hacemos creando un elemento llamado *div* y obteniendo una referencia a este elemento en el DOM (Document Object Model) del navegador.

3. GMap2: El objeto elemental

```
var map = new GMap2(document.getElementById("map_canvas"));
```

La clase de JavaScript que representa el mapa es la clase GMap2. Un objeto de esta clase, define un único mapa dentro de una página. Crearemos un nuevo ejemplar de esta clase usando el operador *new* de JavaScript.

Cuando se crea una nueva instancia de mapa, se especifica en la página, un nodo DOM (normalmente un *div*) como un contenedor para el mapa. Los nodos de HTML son hijos del objeto de JavaScript `document`, y obtenemos referencia a este nodo *div* a través del método `document.getElementById()`.

4. Inicialización del mapa

Una vez creado el mapa, su inicialización consiste en establecer las coordenadas del centro del mapa, así como el nivel del zoom con el queremos visualizarlo.

```
map.setCenter(new GLatLng(37.4419, -122.1419), 13);
```

5. Carga del mapa

Para asegurarnos de la correcta carga del mapa, basta con ejecutar la función que construye el objeto *GMap2* una vez que la sección *<body>* de la página HTML recibe un evento `onload`. De esta forma se evitan los comportamientos impredecibles y se controla cómo y cuando se dibuja el mapa.

```
<body onload="initialize()" onunload="GUnload()">
```

La función `GUnload()` es una función destinada para prevenir pérdidas de memoria (memory leaks).

Una vez presentada de forma breve la forma en la que incluimos el Google Maps API en nuestra página web, se va a proceder a la exposición de forma detallada de los componentes que han sido utilizadas en el desarrollo de esta aplicación. Para más información acerca de componentes, eventos, modos de control y servicios que ofrece Google Maps API se puede consultar la página web <http://code.google.com/apis/maps/>.

Markers



Este componente se utiliza para identificar puntos en el mapa. Su constructor toma un objeto `GLatLng` que establece la posición del marker en el mapa y, de forma opcional, el constructor admite el objeto `GMarkerOptions`.

Los Markers están diseñados para ser interactivos. Por defecto reciben eventos "click", que son a menudo tratados con eventos listeners para desplegar ventanas de información.

En nuestra aplicación los Markers representan la posición en la que queremos ubicar en el mapa cada uno de los sites de los cuales queremos visualizar su monitorización.

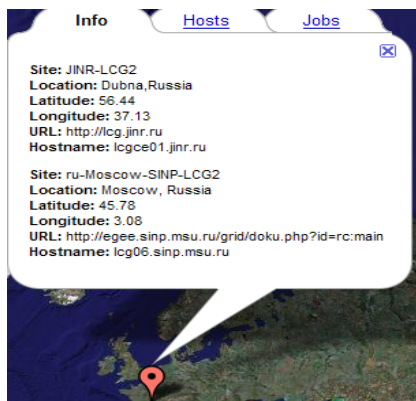
Creación de un Marker

```
point = new GLatLng(vlats[i],vlongs[i]);  
var marker = new GMarker(point);
```

Inclusión del Marker en el mapa

```
map.addOverlay(marker);
```

Tabbed Info Window (Ventana con pestañas de información)



Esta ventana con pestañas de información, es la que usamos en la aplicación para organizar la información de cada *SiteGroup*. Para cada *SiteGroup* se despliega una ventana con tres pestañas de información. La primera, muestra información sobre cada uno de los *sites* del grupo, y las otras dos muestran información acerca de la ejecución de los comandos *gwhoost* y *gwps* respectivamente.

El despliegue de esta ventana está asociado al evento "click" asociado al marker. Pasamos un array de instancias *tabs* de esta clase al método

`marker.openInfoWindowTabsHtml(tabs)`. Si este array tiene más de un elemento, entonces la ventana será mostrada con tantas pestañas como número de elementos tenga. En el constructor se introduce tanto el texto correspondiente a cada pestaña, como el contenido html que se mostrara en cada una de las mismas.

En el fragmento de código siguiente, observamos como creamos este componente en nuestra aplicación:

```
// Función que crea una ventana con pestañas

function createTabbedMarker(point,htmls,labels,name) {
    var marker = new GMarker(point);
    GEvent.addListener(marker, "click", function() {
        if (htmls.length > 2)
            htmls[0]="<div style='width:" + htmls.length*88 + "px'>" +
htmls[0] + "</div>";
        var tabs = [];
        for (var i=0; i<htmls.length; i++)
            tabs.push(new GInfoWindowTab(labels[i],htmls[i]));
        marker.openInfoWindowTabsHtml(tabs);
        markerWindowOpen = marker;
    }
};
```

ELabel



El componente Elabel se trata de una extensión de Google Maps API que permite colocar etiquetas, imágenes, o contenido HTML. En nuestra aplicación, este ha sido el componente que usamos para visualizar la barra con porcentaje enmarcado y absolutos con porcentaje entre paréntesis correspondiente a la métrica especificada en la ventana de configuración.

Para la inclusión de este componente en la aplicación hemos seguido los siguientes pasos:

1. Crear una CSS Class Style para las Elabels

```
/* Estilo de los elabels */

.style0 {

    background-color:#ffffff;

    font-weight:bold;

    border:2px #006699 solid;}
```

2. Incluir en el directorio de la aplicación el fichero *elabel.js*

```
<script src="../../20_4v2/elabel.js" type="text/javascript"></script>
```

3. Crear los objetos *elabel* e incluirlos en el mapa

```
// Creamos las etiquetas que mostraran el volumen de trabajo sobre  
cada una de las ciudades  
for (i=0; i<vsimtrab.length; i++){  
    vElabels[i] = new ELabel(points[i],vsimtrab[i],"style0",new  
GSize(-40,40),60);  
    map.addOverlay(vElabels[i]);  
}
```

Para ir conociendo en todo momento en todo momento las coordenadas sobre las cuales estamos moviendo el ratón por el mapa, así como el punto sobre el que este está centrado y el nivel de zoom que se está aplicando. Hemos añadido una ventana informativa.

Mouse Position	
Latitude	Longitude
50.0641917	35.1562500
Center Map	
32.8426736	34.4531250
Zoom Map	
2	

Todos estos datos se añaden a la tabla en el listener que reacciona al evento “*mousemove*”:

```
var lastPoint;  
  
GEvent.addListener(map, "mousemove", function(point){  
    var center = map.getCenter();  
    var xcent = center.lat().toFixed(14).substr(0,10) ;  
    var ycent = center.lng().toFixed(14).substr(0,10) ;  
    var zoom = map.getZoom();  
    var latStrF = point.lat().toFixed(14).substr(0,10) ;  
    var longStrF = point.lng().toFixed(14).substr(0,10) ;  
  
    document.getElementById("latmouse").innerHTML = latStrF;  
    document.getElementById("longmouse").innerHTML = longStrF;  
    document.getElementById("xcenter").innerHTML = xcent;  
    document.getElementById("ycenter").innerHTML = ycent;  
    document.getElementById("zoom").innerHTML = zoom;  
    lastPoint = point;  
});
```

8.1.3. HTML

Es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

HTML se escribe en forma de "etiquetas", rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo JavaScript), el cual puede afectar al comportamiento de navegadores web y otros procesadores de HTML.

En nuestra aplicación el documento HTML es el encargado de la visualización web de la misma. En él se muestra el mapa, la ventana de configuración, y el diagrama de barras. Desde él y usando JavaScript se lee toda la información a visualizar desde archivos en formato XML.

Un documento HTML se divide en dos grandes bloques:

1. El **bloque <head>**: define la cabecera del documento HTML. Esta cabecera suele contener información sobre el documento que no se muestra directamente al usuario. Como por ejemplo el título de la ventana del navegador. En él, nosotros hemos declarado las variables globales y todas las funciones que usamos para el desarrollo.
2. El **bloque <body>**: define el contenido principal o cuerpo del documento. Esta es la parte del documento html que se muestra en el navegador; dentro de esta etiqueta pueden definirse propiedades comunes a toda la página, como color de fondo y márgenes. Dentro del cuerpo <body> podemos encontrar numerosas etiquetas.

En este bloque, ayudándonos del software *Dreamweaver* (programa muy utilizado en la programación y diseño web), para facilitar nuestra tarea y así no tener que escribir a mano todas las etiquetas, hemos incluido todos elementos de nuestra aplicación (tablas, divs, títulos, list/menu, etc).

También se incluye en este bloque todo el código JavaScript que gestiona el funcionamiento dinámico de la página.

8.1.4. XML

XML es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML pero su función principal es describir datos y no mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones.

XML se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.

En nuestra aplicación tenemos los siguientes ficheros xml:

1. Fichero de Agrupación

Por cada site tenemos un fichero de agrupación que recoge el conjunto de host que forman parte del site.

Su estructura es la siguiente:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<fichAgrupacion>
  <elemagrupacion>
    <Site> </Site>
    <Host> </Host>
  </elemagrupacion>
  <elemagrupacion>
    <Site> </Site>
    <Host> </Host>
  </elemagrupacion>
  .
  .
  .
</fichAgrupacion>
```

2. xmlgwhost.xml

Recoge toda la información correspondiente a la ejecución del comando gwhost.

Su estructura es la siguiente:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<ghostxml>
  <HOST>
    <OS></OS>
    <HOSTNAME> </HOSTNAME>
    <ARCH> </ARCH>
    <N>
      <U></U>
      <F></F>
      <T></T>
    </N>
    <LRMS> </LRMS>
  </HOST>
  <HOST>
    <OS></OS>
    <HOSTNAME> </HOSTNAME>
    <ARCH> </ARCH>
    <N>
      <U></U>
      <F></F>
      <T></T>
    </N>
    <LRMS> </LRMS>
  </HOST>
  .
```

```
.  
.  
</ghostxml>
```

3. *xmlgwps.xml*

Recoge toda la información correspondiente a la ejecución del comando gwps.

Su estructura es la siguiente:

```
<?xml version="1.0" encoding="iso-8859-1"?>  
<ghostxml>  
  <HOST>  
    <HOSTNAME> </HOSTNAME>  
    <PENDING></PENDING>  
    <EXECUTING></EXECUTING>  
    <NUMBER_USERS></NUMBER_USERS>  
  </HOST>  
  <HOST>  
    <HOSTNAME> </HOSTNAME>  
    <PENDING></PENDING>  
    <EXECUTING></EXECUTING>  
    <NUMBER_USERS></NUMBER_USERS>  
  </HOST>  
  .  
  .  
  .  
</ghostxml>
```

8.1.5. Perl

Perl es un lenguaje de propósito general originalmente desarrollado para la manipulación de texto y que ahora es utilizado para un amplio rango de tareas incluyendo, administración de tareas, desarrollo web, programación en red, desarrollo de GUI y más.

Se previó que fuera práctico (facilidad de uso, eficiente, completo) en lugar de hermoso (pequeño, elegante, mínimo). Sus principales características son que es fácil de usar, soporta tanto la programación estructurada como la programación orientada a objetos y la programación funcional, tiene incorporado un poderoso sistema de procesamiento de texto y una enorme colección de módulos disponibles.

En el desarrollo de nuestra aplicación nos hemos valido de ficheros perl para, a partir de la salida de los comandos gwhost y gwps, que volcamos a ficheros de texto, generar los ficheros XML que recojan esta información y pueda así ser leída por el script del HTML.

8.1.6. JavaScript

JavaScript es un lenguaje de scripts desarrollado por Netscape para incrementar las funcionalidades del lenguaje HTML. Sus características más importantes son:

1. JavaScript es un lenguaje interpretado, es decir, no requiere compilación. El navegador del usuario se encarga de interpretar las sentencias JavaScript contenidas en una página HTML y ejecutarlas adecuadamente.
2. JavaScript es un lenguaje orientado a eventos. Cuando un usuario pincha sobre un enlace o mueve el puntero sobre una imagen se produce un evento. Mediante JavaScript se pueden desarrollar scripts que ejecuten acciones en respuesta a estos eventos.
3. JavaScript es un lenguaje orientado a objetos. El modelo de objetos de JavaScript está reducido y simplificado, pero incluye los elementos necesarios para que los scripts puedan acceder a la información de una página y puedan actuar sobre la interfaz del navegador.

En nuestra aplicación hemos incluido JavaScript de las dos formas posibles en las que esto se puede hacer:

1. Embebido en el código HTML, entre las etiquetas o tags `<script>` y `</script>`.
2. Como archivo .js que se carga con la página HTML. Para ello, debe indicarse en las tags anteriores el nombre y ubicación del archivo .js que contiene el script JavaScript.

```
<script src="../../20_4v2/elabel.js" type="text/javascript"></script>  
<script src="../../20_4v2/diagram.js" type="text/javascript"></script>
```

La librería diagram.js es especialmente útil para mostrar un diagrama de barras dependiente de la entrada de datos que le introduzca el usuario o de datos presentes en una base de datos.

Para la construcción de este diagrama hay que tener en cuenta dos objetos:

1. Diagram object

Este objeto ofrece la posibilidad de dibujar un diagrama en una posición definida con un tamaño definido dentro del tamaño del documento.

Esta es la forma en la que hemos definido el diagrama en nuestra aplicación:

```
//Se indica la posición del diagrama en el documentos  
D.SetFrame(45, 575, 685, 575 + (vsimtrab.length*55) );  
  
//Indica el rango de valores en los ejes x e y de la gráfica  
D.SetBorder( 0, 100, -1, vsimtrab.length);  
  
//Modifica el estilo del texto  
D.Font="font-family:Verdana;font-weight:bold;font-size:11pt;line-  
height:15pt;line-width:15pt;";
```

```
//Pone título al diagrama
D.SetText("", "", "Performance Graph");
D.YScale= 0;
D.XScale=" %";
D.SetGridColor("#8CACBB", "#CCCCCC");
D.Draw("#DEE7EC", "#000000", false);
```

2. The Box object

Este objeto lo usamos para mostrar una barra en la pantalla. Más en concreto en el diagrama. Con el propósito de encontrar la posición y el tamaño apropiado de la barra. Es posible mover, redimensionar, y borrar una barra después de haber sido dibujada.

Esta es la forma en la que creamos las barras en nuestra aplicación:

```
for (i=0; i<vsimtrab.length; i++){
    vj[i]=D.ScreenY(i+0,5);
    bottom= D.ScreenX(0)

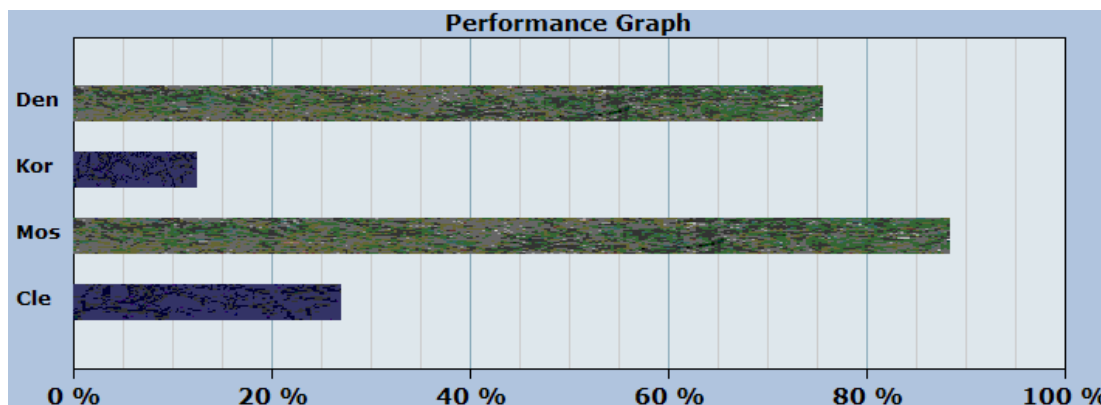
    if (i%2==0) {
        vBars[i] = new Box(bottom,vj[i]+12, D.ScreenX(vporciento[i]),
vj[i]-12, "#333366","mar.gif", "#FFFFFF","", "#000000");
        vBars2[i] = new Bar(bottom - 37,vj[i]-12,
D.ScreenX(vporciento[i]), vj[i]-12, "", " ", "#000000","");
    }
    else {
        vBars[i] = new Box(bottom,vj[i]+12, D.ScreenX(vporciento[i]),
vj[i]-12, "#8894A4","verde.gif", "#FFFFFF","", "#000000");
        vBars2[i] = new Bar(bottom - 37,vj[i]-12,
D.ScreenX(vporciento[i]), vj[i]-12, "", " ", "#000000","");
    }
}
```

Vamos creando una barra de cada color de forma alternativa.

El fragmento de código donde las barras se redimensionan para mostrar los datos actualizados en tiempo real es el siguiente:

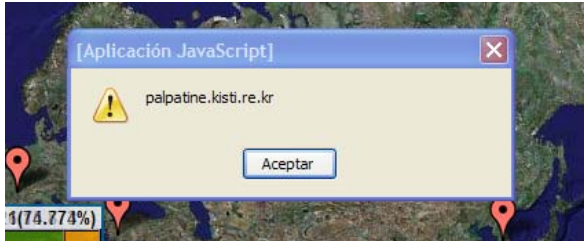
```
vBars[i].ResizeTo(bottom,vj[i]-12,D.ScreenX(vporciento[i])-bottom,24);
```

Este es el aspecto que presenta el diagrama de barras de nuestra aplicación:



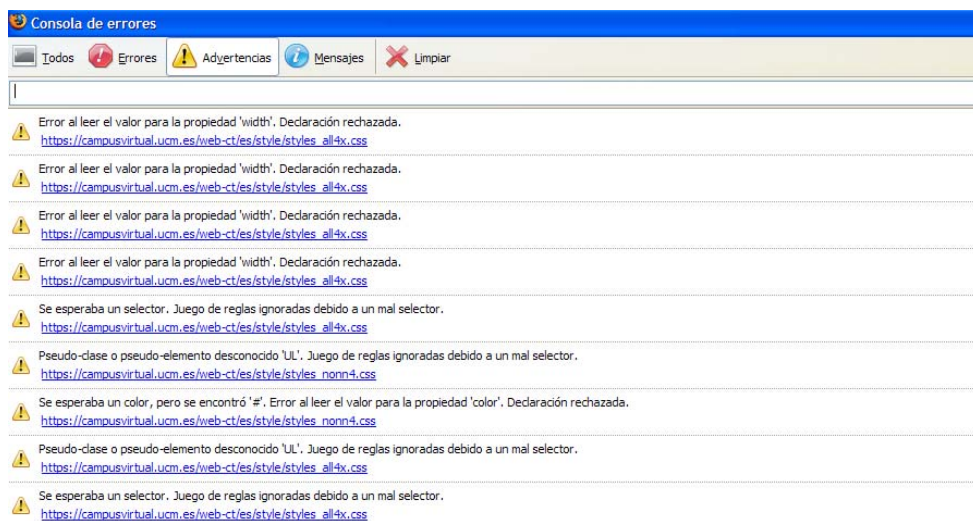
Depuración

Para hacer la depuración de nuestro código JavaScript, en un principio lo hacíamos a través de `alerts()` que muestran mensajes por pantalla y continua la ejecución una vez que pulsas aceptar.



Sin embargo, Firefox nos ofrece un método mucho más cómodo y conveniente, permitiéndonos escribir los mensajes en una consola mediante el método `dump`. Para ello tenemos que activar primero en las opciones de configuración, el uso de la función `dump`, abriendo `about:config` y creando una nueva entrada de tipo lógico con nombre `browser.dom.window.dump.enabled` y valor `true`.

Una vez hecho esto, al abrir Firefox en modo consola de error podremos utilizar `dump(cadena)` para escribir texto en la consola.



8.1.7. AJAX

Como ya mencionamos en el apartado de introducción de Google Maps, para conseguir la conectividad sin sincronía con el servidor y así proporcionar al usuario mayor interactividad con el mapa, se usa una tecnología bastante reciente, denominada AJAX, la cual vamos a describir brevemente en este apartado.

Asynchronous JavaScript And XML (JavaScript asíncrono y XML) o AJAX, es una técnica de desarrollo web creada con el fin de crear aplicaciones interactivas. Estas se ejecutan en el navegador de los usuarios y mantienen comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma

página sin necesidad de refrescarla. Lo que significa aumentar la interactividad, velocidad y usabilidad en la misma

AJAX es una combinación de cuatro tecnologías ya existentes:

- **XHTML**(o **HTML**) y hojas de estilo en cascada (CSS) para el diseño que acompaña a la información
- Document (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- El objeto **XMLHttpRequest** para intercambiar datos asincrónicamente con el servidor web.
- XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano, JSON y hasta EBML.

Como el DHTML, LAMP o SPA, AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente.

Gracias a esta tecnología, el código JavaScript de nuestra aplicación lee la información de entrada que se encuentra en formato XML. Debido al asincronismo en un principio tuvimos problemas con la ejecución del script, ya que este no se trataba de igual forma en los distintos navegadores, y en ocasiones se trataba de acceder a objetos que aún no habían sido creados.

8.1.8. Uso de GridWay en la aplicación

Monitorización de trabajos

Una de las principales características de la aplicación desarrollada, es la monitorización en tiempo real de ciertos datos correspondientes a los trabajos y a los hosts que se encuentran en la red Grid.

Para esta monitorización existirán dos ficheros que muestran en todo momento la información, tanto del estado de los trabajos, como de los Hosts que conforman la red.

Para llevar a cabo esta monitorización, se realiza una conexión a la red grid, y se llama a las funciones **gwps**, que devuelve el historial de trabajos lanzados en la red, y **gwhost**, para ver los recursos disponibles.

Formato de la salida correspondiente a la ejecución del comando **gwps**:

```
$ gwps
USER          JID DM  EM   START    END      EXEC    XFER    EXIT NAME  HOST
gwtutorial00  0  done ---- 20:16:28 20:18:16 0:00:55 0:00:08 0   stdin aquila.dacya.ucm.es/SGE
tinova        1  done ---- 12:26:46 12:31:15 0:03:55 0:00:08 0   stdin hydrus.dacya.ucm.es/PBS
tinova        2  pend ---- 12:38:38 ---:---: 0:00:00 0:00:00 --  t.jt  --
```

Formato de la salida correspondiente a la ejecución del comando **gwhost**:

HID	PRI	OS	ARCH	MHZ	%CPU	MEM(F/T)	DISK(F/T)	N(U/F/T)	LRMS	HOSTNAME
0	1	Linux2.6.17-2-6	x86	3216	0	44/2027	76742/118812	0/0/2	Fork	cygnus.dacya.ucm.es
1	1			0	0	0/0	0/0	0/0/0		orion.dacya.ucm.es
2	1	Linux2.6.18-4-a	x86_6	2211	100	819/1003	77083/77844	0/2/4	PBS	hydrus.dacya.ucm.es
3	1	Linux2.6.17-2-6	x86	3216	163	1393/2027	101257/118812	0/2/2	Fork	draco.dacya.ucm.es
4	1	Linux2.6.18-4-a	x86_6	2211	66	943/1003	72485/77844	0/5/5	SGE	aquila.dacya.ucm.es

De las salidas de estos dos comandos es de donde nuestra aplicación obtiene los datos que usa para generar las estadísticas que se quieren mostrar.
Estas son las siguientes:

Free slots: F(gwhost)/T(gwhost)

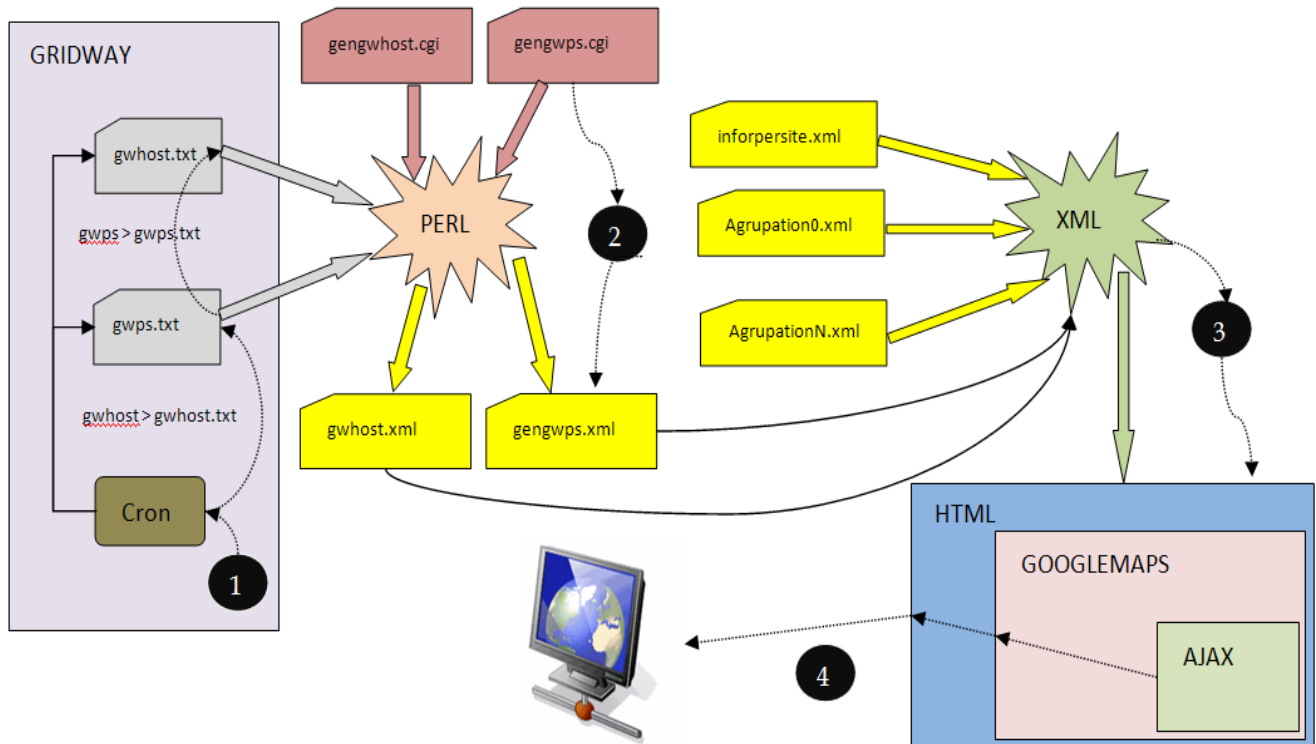
Used slots: U(gwhost)/T(gwhost)

Pending jobs: Pending(gwps)/Executing(gwps)+Pending(gwps)

Executing jobs: Executing(gwps)/Executing(gwps)+Pending(gwps)

8.1.9. Relación entre las tecnologías:

El siguiente diagrama muestra la relación, paso a paso, entre las distintas que conforman la aplicación:



- 1) El demonio o cron, se encarga de ejecutar periódicamente los comandos de Gridway `gwhost` y `gwps`, y generar dos ficheros de texto (`gwhost.txt` y `gwps.txt`) con la salida de cada uno de ellos.
- 2) La información contenida en `gwhost.txt` y `gwps.txt` es transformada a formato XML por los programas PERL `gengwhost.cgi` y `gengwps.cgi` respectivamente, y depositada en los ficheros `gwhost.xml` y `gwps.xml`.
- 3) La información de estos dos ficheros, que se actualiza dinámicamente cada cierto tiempo, junto con la información contenida en otros ficheros estáticos (`agrupation0.xml`, `agrupation1.xml`, `inforpersite.xml`) es recogida por `aplicación.html`.
- 4) Es en `aplicación.html` y gracias al uso de JavaScript, GoogleMapsAPI y AJAX, donde se recogen los datos y se gestiona la visualización de los mismos por el usuario a través de la web.

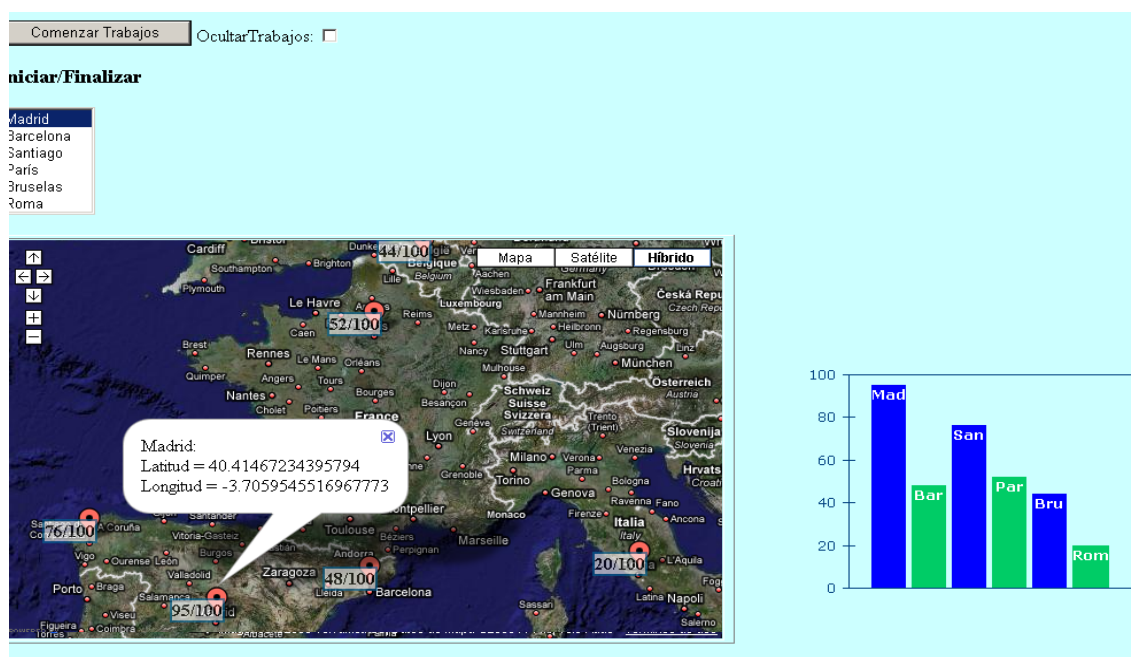
8.2. Evolución del proyecto

La realización del proyecto se llevó a cabo mediante la implementación sucesiva de prototipos, cada uno de los cuales aportaba alguna novedad a los anteriores. A continuación, se muestran los principales prototipos que se realizaron, explicando para cada uno de ellos en que fecha (aproximadamente) fue realizado, sus principales características, las mejoras respecto a las versiones anteriores y los fallos corregidos en versiones sucesivas.

Antes de empezar el repaso por los principales prototipos, es necesario destacar que se puede apreciar que en ocasiones se realizan varios prototipos en poco tiempo y, en otros casos, pasa incluso más de un mes sin que haya novedades. Esto se debe a que no siempre nuestra dedicación al proyecto fue la misma (hubo periodos como las Navidades o el periodo de exámenes en los que se tuvo que reducir el tiempo dedicado al proyecto); además, en ocasiones los problemas a solucionar o las mejoras a realizar fueron más difíciles de implementar, suponiendo un mayor tiempo de trabajo para terminar le siguiente prototipo.

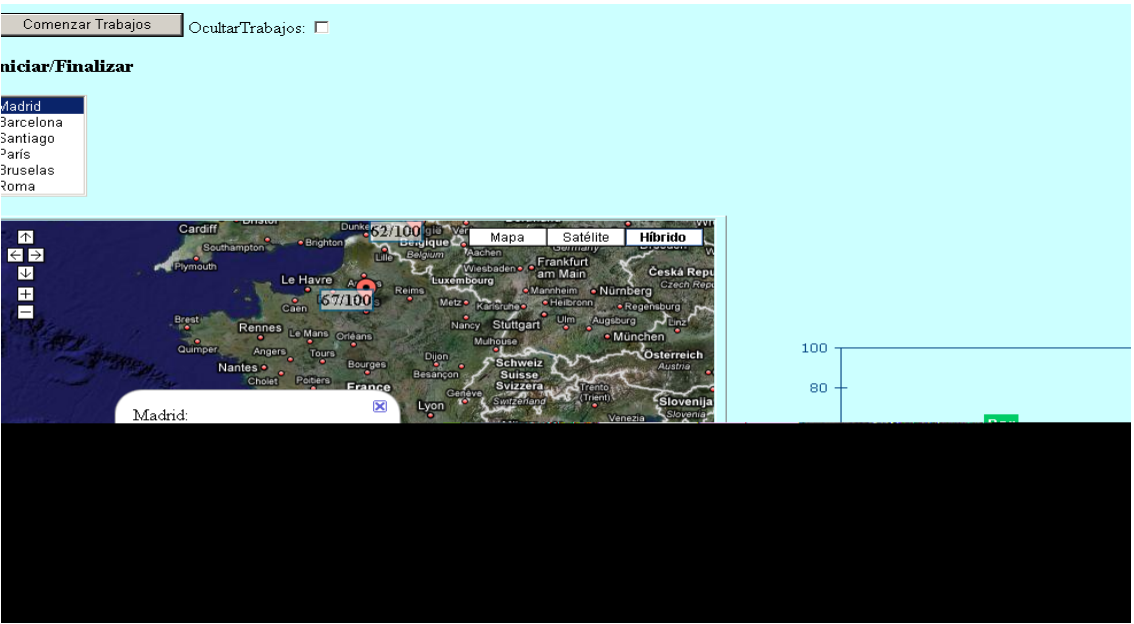
8.2.1. Prototipo 1 (10 de Diciembre de 2007)

Realmente, ya se habían realizado algunos prototipos antes que éste; sin embargo, es el primero que mencionamos ya que supone el primer prototipo en el que se aprecian algunas de las características que se mantendrían hasta la versión definitiva. Entre estas características destacan el mapa y la gráfica como elementos principales de la aplicación, incluyendo el mapa los elabels para mostrar los datos de cada host. No obstante, se aprecia la falta de opciones de configuración (la única que permite es seleccionar que trabajos se desea que se muestren) y un aspecto de la página poco profesional; además, destaca el importante problema de que la aplicación no es compatible con el navegador Firefox (problema que se mantendría durante bastantes prototipos posteriores a éste).



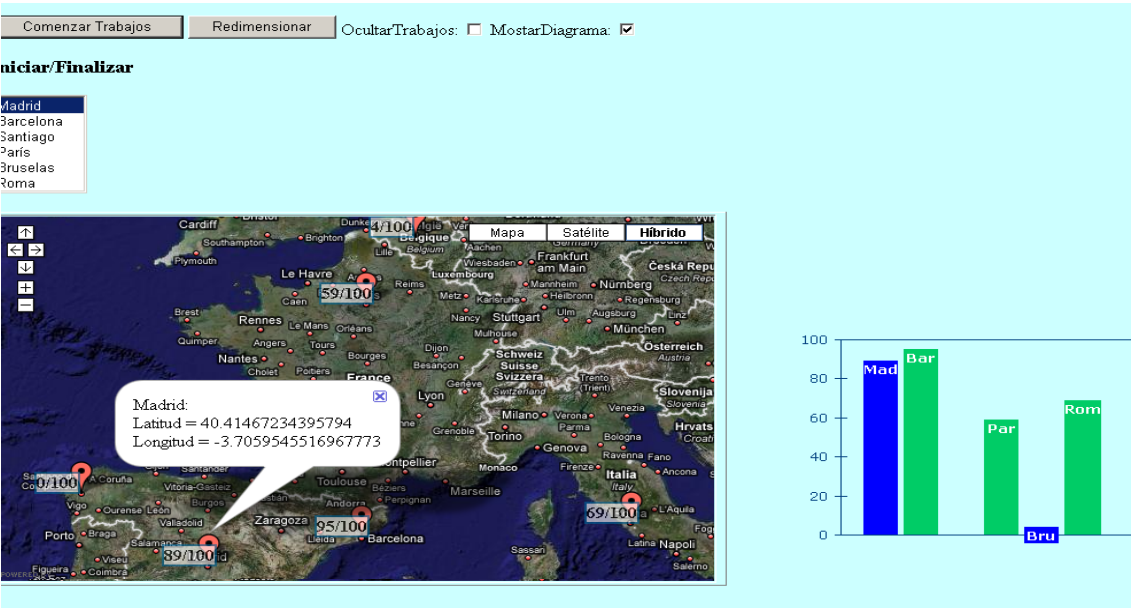
8.2.2. Prototipo 2 (11 de Diciembre de 2007)

Presenta como novedad el hecho de que las ventanas de monitorización muestran, además de los datos que ya se mostraban en la versión anterior (nombre de la ciudad y coordenadas de ésta), la carga de trabajo que hay al pinchar en el marcador. Sin embargo, dicha carga se mantiene estática en el marcador (a pesar de que tanto en el mapa como en la gráfica si se va modificando dicho valor).



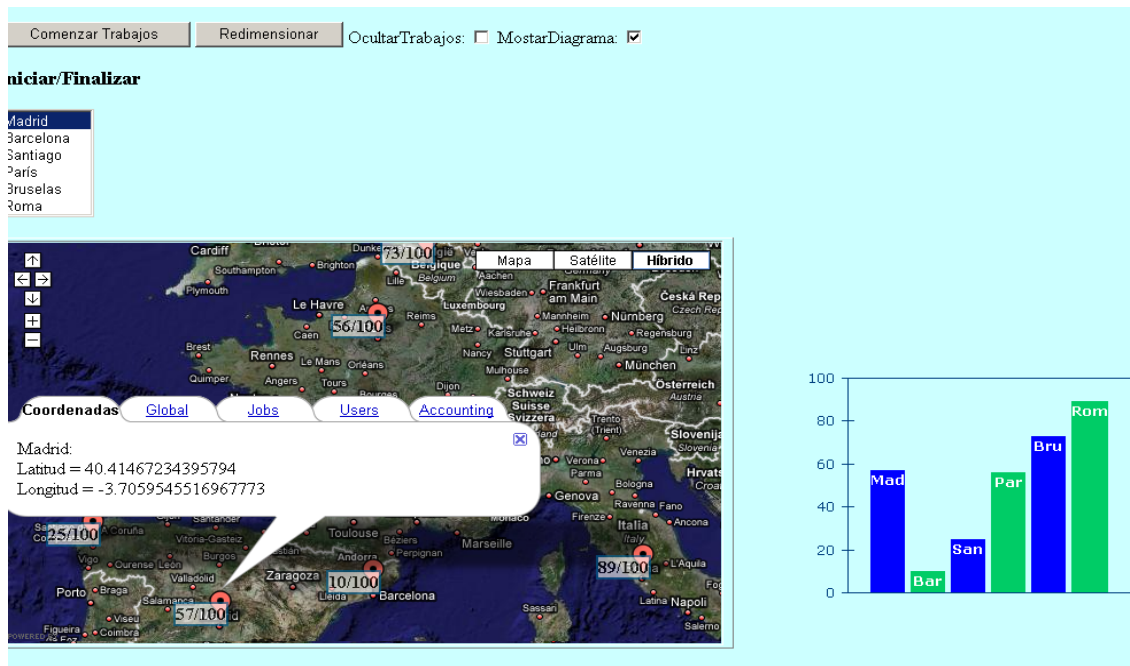
8.2.3. Prototipo 3 (12 de Diciembre de 2007)

En este prototipo ya empiezan a aparecer las primeras opciones de configuración. De esta manera, a partir de esta versión resulta posible redimensionar el mapa, además de poder elegir si se quiere que se muestre o no la gráfica.



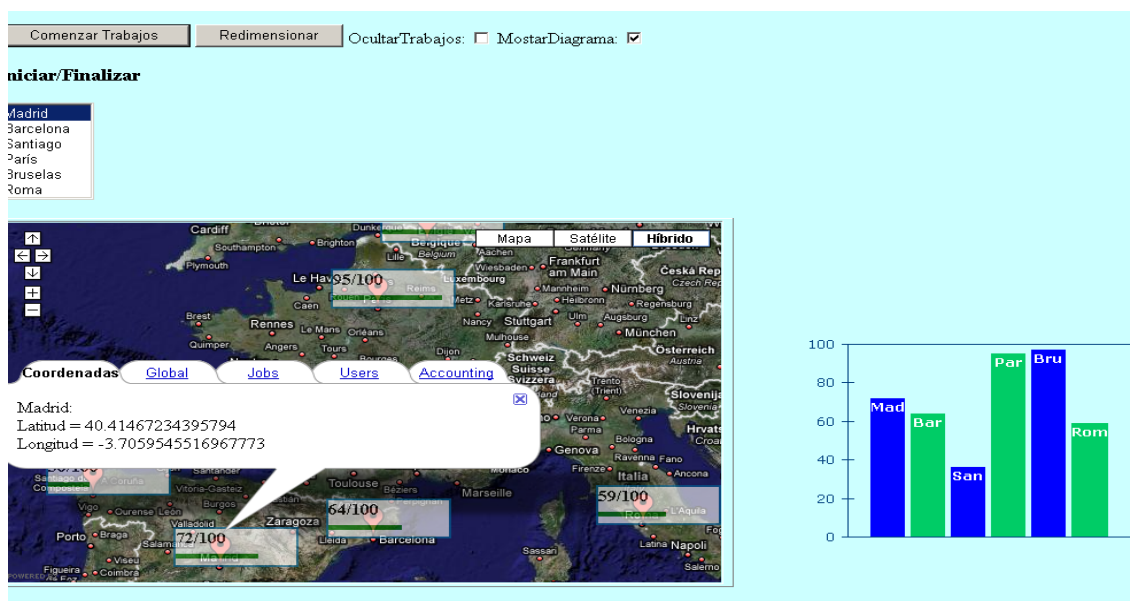
8.2.4. Prototipo 4 (17 de Diciembre de 2007)

Presenta como novedad el que se añaden distintas pestañas a las ventanas de monitorización, si bien la mayoría de estas pestañas no presentan ningún contenido. Además, la carga de trabajo que se muestra en éstas se actualiza dinámicamente.



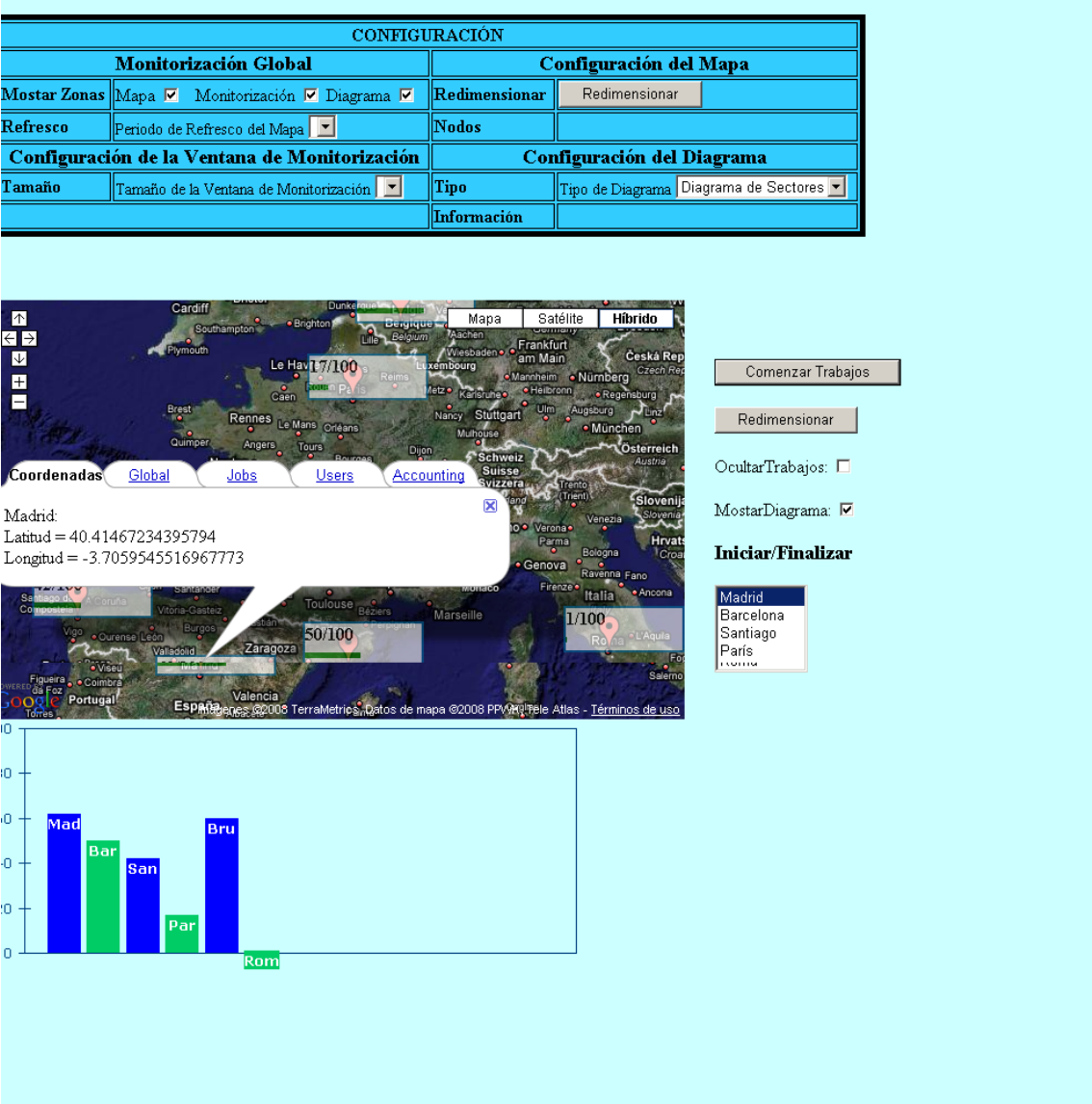
8.2.5. Prototipo 5 (3 de Enero de 2008)

La mejora introducida consiste en que los elabels pasan a mostrar la carga de trabajo no sólo de forma numérica sino también con una gráfica esquemática.



8.2.6. Prototipo 6 (15 de Enero de 2008)

Presenta como principal novedad la aparición del recuadro de configuración en la parte superior de la página; sin embargo, la mayoría de las opciones que se muestran en él no están implementadas. Por otro lado, se modifica la posición del diagrama, pasando de estar a la derecha del mapa a estar debajo.



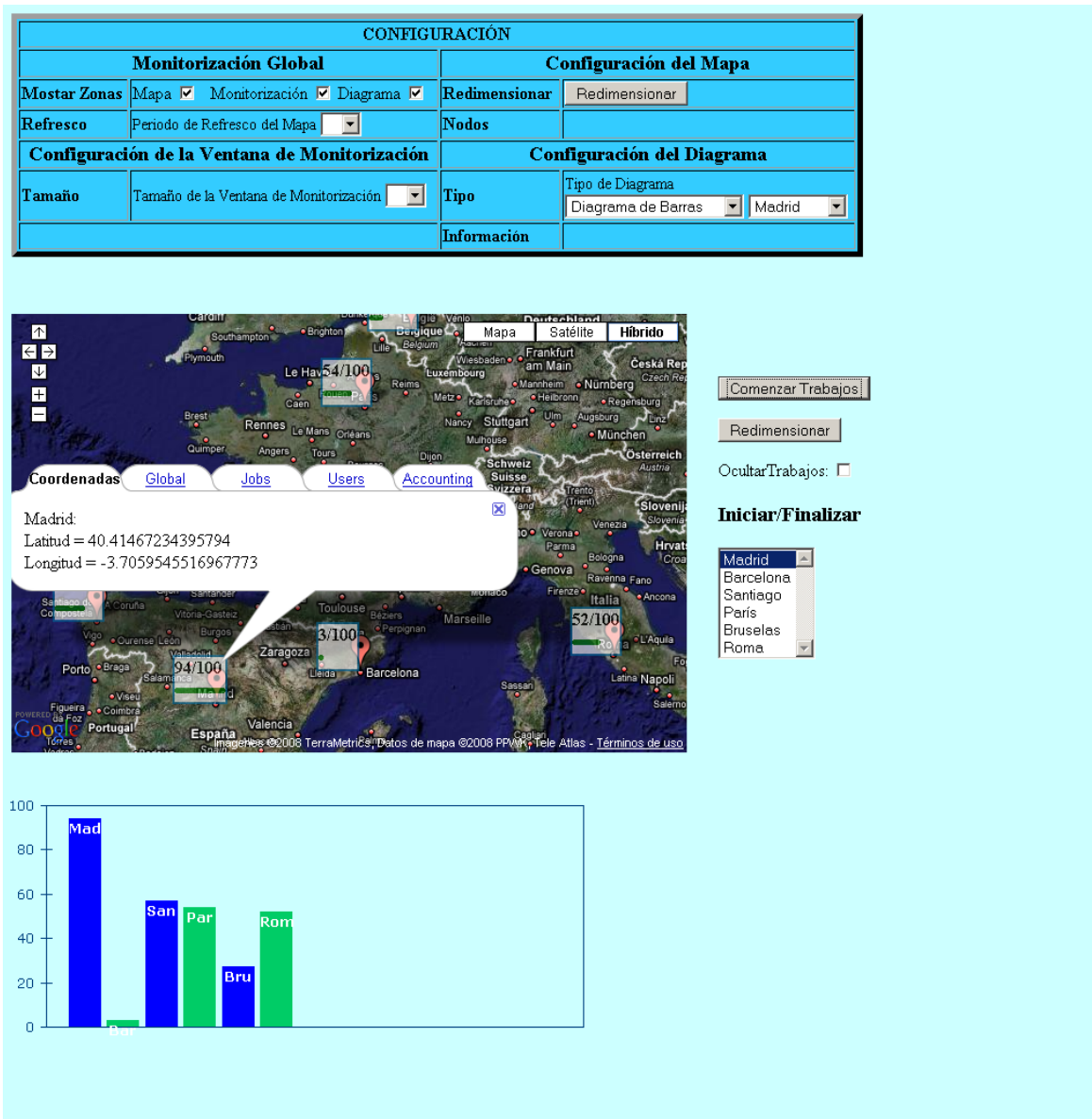
8.2.7. Prototipo 7 (16 de Enero de 2008)

Este prototipo supuso un hito ya que fue el primero que conseguimos que funcionase correctamente con el navegador Firefox (problema que nos llevó arreglar algo más de un mes). Por otra parte, se añadió un recuadro de monitorización que mostraba datos de cada uno de los host, pero el hecho de que aportaba poco a la aplicación (pues sólo mostraba datos que ya se mostraban en otros elementos de ésta) y las dificultades intrínsecas a su implementación hicieron que este recuadro acabase desapareciendo de nuestro proyecto (de hecho, ya fue eliminado en el siguiente prototipo).



8.2.8. Prototipo 8 (20 de Enero de 2008)

Como ya hemos comentado, en esta versión desaparece el recuadro de monitorización añadido al prototipo anterior. Además, se añade la opción de poder seleccionar el tipo de gráfica a visualizar (diagrama de barras con las cargas de trabajo de todas las ciudades o diagrama de sectores con la carga de trabajos de una sola ciudad); no obstante, más adelante se optó por prescindir del diagrama de sectores, pues el diagrama de barras parecía más apropiado para la tarea de monitorización.

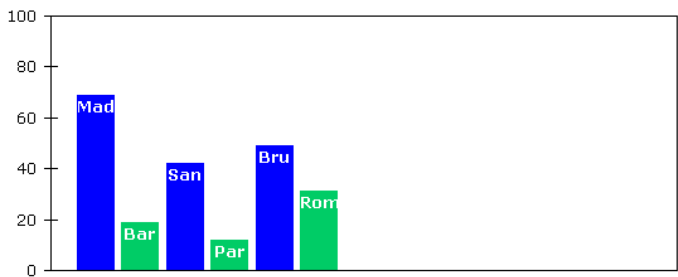
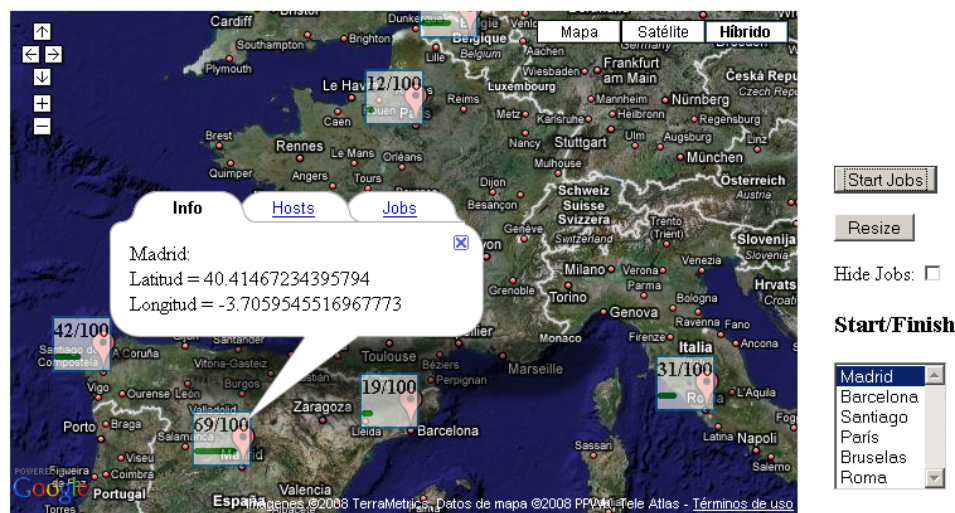


8.2.9. Prototipo 9 (27 de Febrero de 2008)

Entre las novedades se encuentran diversas modificaciones relativas a la profesionalización del aspecto de la aplicación, como son cambiar los colores de ésta o traducir su contenido textual al inglés. Además, se realizan importantes cambios en el recuadrado de configuración, de forma que se le añaden botones de minimización y maximización, se modifican las opciones presentes en éste y se implementan las opciones que aparecen ahora en el recuadro con la excepción del elemento “Metric”. Por último, también se modifica el contenido de las ventanas de monitorización, de forma que estas pasan de tener cinco pestañas a tener las tres pestañas de la versión definitiva.

CONFIGURATION

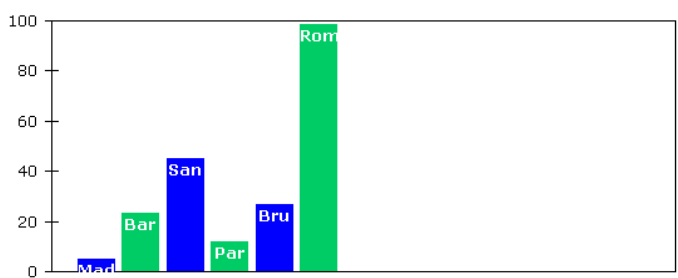
Global Monitorization		Infrastructure Map Configuration	
Visible Elements	Infrastructure Map <input checked="" type="checkbox"/> Performance Graph <input checked="" type="checkbox"/>	Resize	<button>Resize</button>
Refresh Period	Refresh Period 5	Sites	Madrid Barcelona
Performance Graph Configuration		Performance Metric	
Type	Diagram Type Bar Diagram Madrid	Metric	Free Slots



8.2.10. Prototipo 10 (3 de Marzo de 2008)

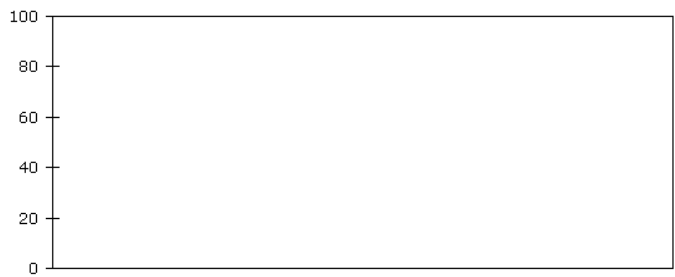
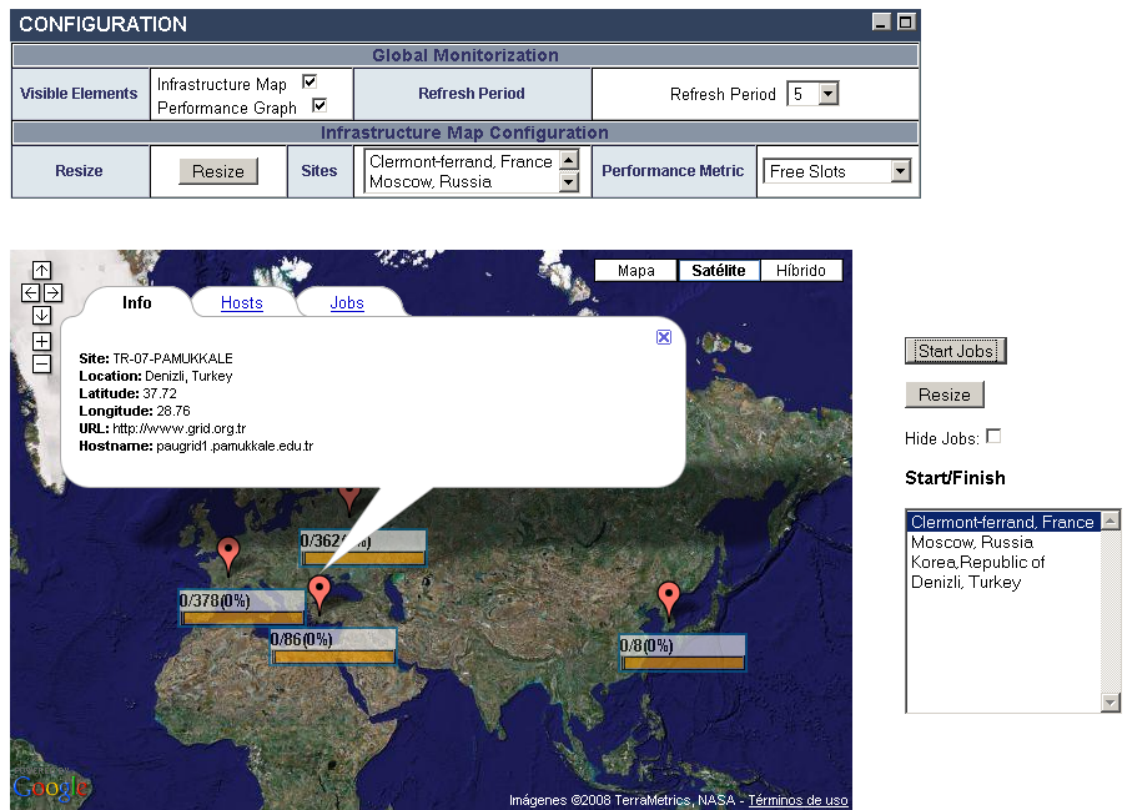
Se modifica el formato de la información que aparece en los elabels. De esta forma, ahora las gráficas de los elabels se visualizan con un formato más estilizado, y en cada elabel aparece no sólo el porcentaje (como aparecía en los prototipos anteriores) sino también el número de trabajos activos y el número de trabajos totales de cada sitio.

CONFIGURATION			
Global Monitorization		Infrastructure Map Configuration	
Visible Elements	Infrastructure Map <input checked="" type="checkbox"/> Performance Graph <input checked="" type="checkbox"/>	Resize	<input type="button" value="Resize"/>
Refresh Period	Refresh Period <input type="text" value="5"/>	Sites	<input type="text" value="Madrid"/> <input type="text" value="Barcelona"/>
Performance Graph Configuration		Performance Metric	
Type	Diagram Type <input type="text" value="Bar Diagram"/> <input type="text" value="Madrid"/>	Metric	<input type="text" value="Free Slots"/>



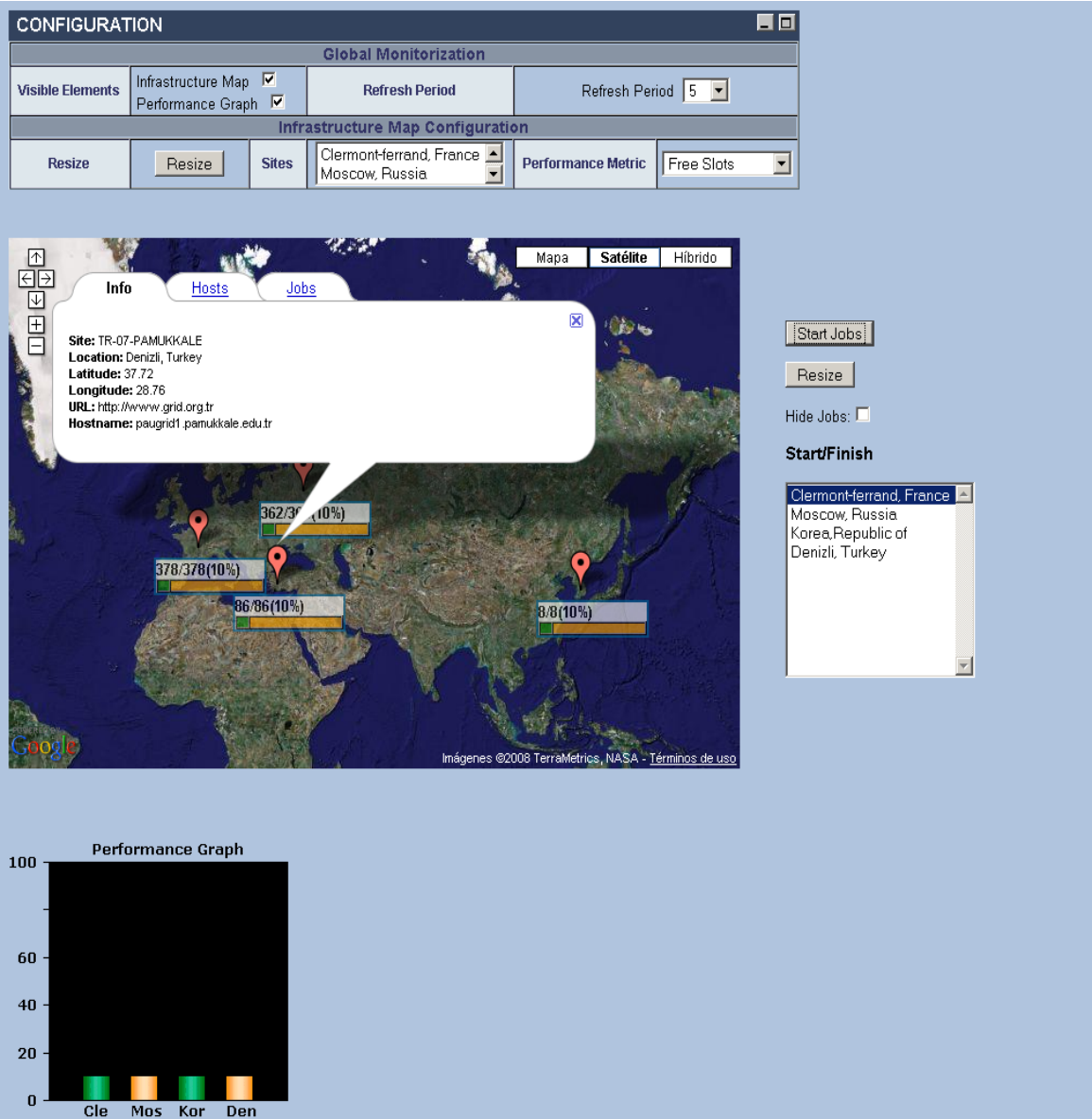
8.2.11. Prototipo 11 (2 de Abril de 2008)

Se altera el formato del recuadro de configuración eliminando algunas opciones (como la de elegir el tipo de gráfica) y cambiando su apariencia visual. Por otro lado, se crea una hoja de estilos externa al fichero principal para poder hacer las modificaciones del estilo de la aplicación de una manera más cómoda y rápida. Finalmente, se pasa a trabajar leyendo los datos de los ficheros xml generados con la invocación de los comandos “gwhost” y “gwps”.



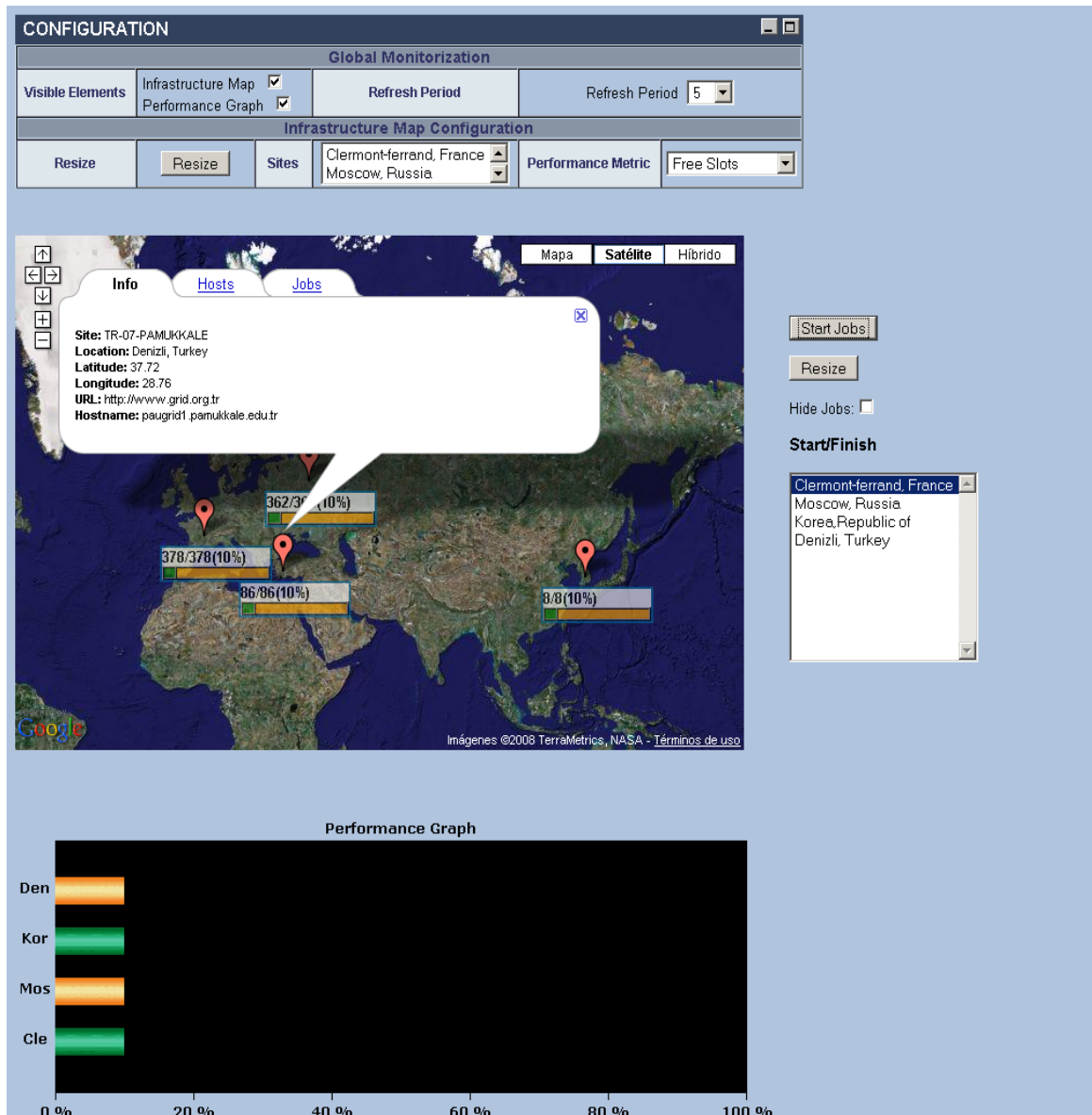
8.2.12. Prototipo 12 (13 de Abril de 2008)

Se realizan modificaciones en el aspecto de la aplicación. Entre estas modificaciones, destaca el cambio que sufre el diagrama, que adquiere una apariencia más acorde con la del resto del prototipo.



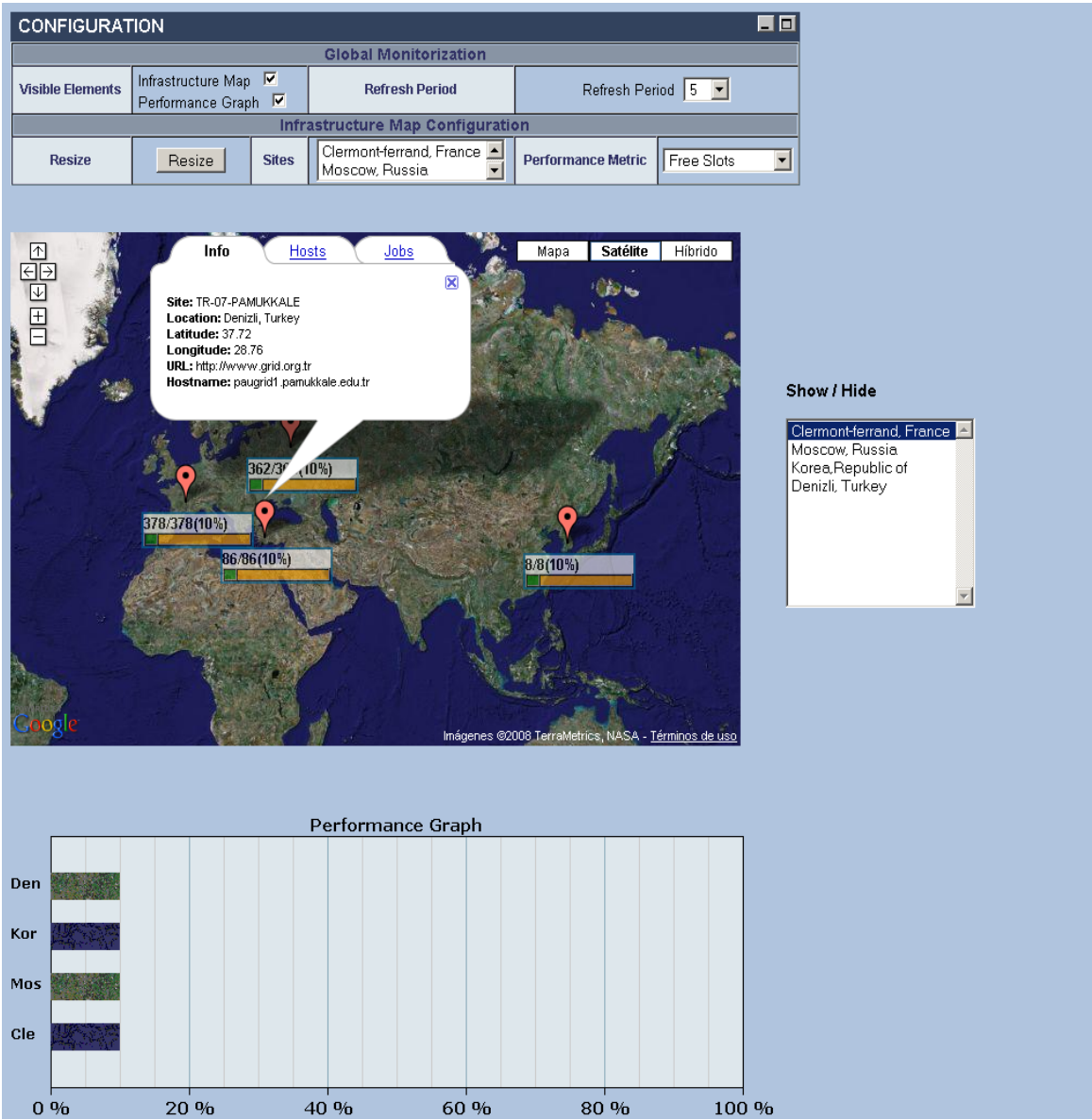
8.2.13. Prototipo 13 (19 de Abril de 2008)

Presenta como novedad la alteración de la gráfica, cuyas barras pasan a estar orientadas verticalmente; ésto supone una mejora ya que permite que si aumenta el número de sitios a monitorizar no aumente la anchura de la aplicación sino que lo que aumente sea su longitud, lo que sin duda supone una solución mucho más elegante. Por otro lado, se modifican los recuadros con la lista de sitios de forma que cuando se oculta un trabajo aparezca al lado del sitio correspondiente la palabra “hidden” para indicar su estado.



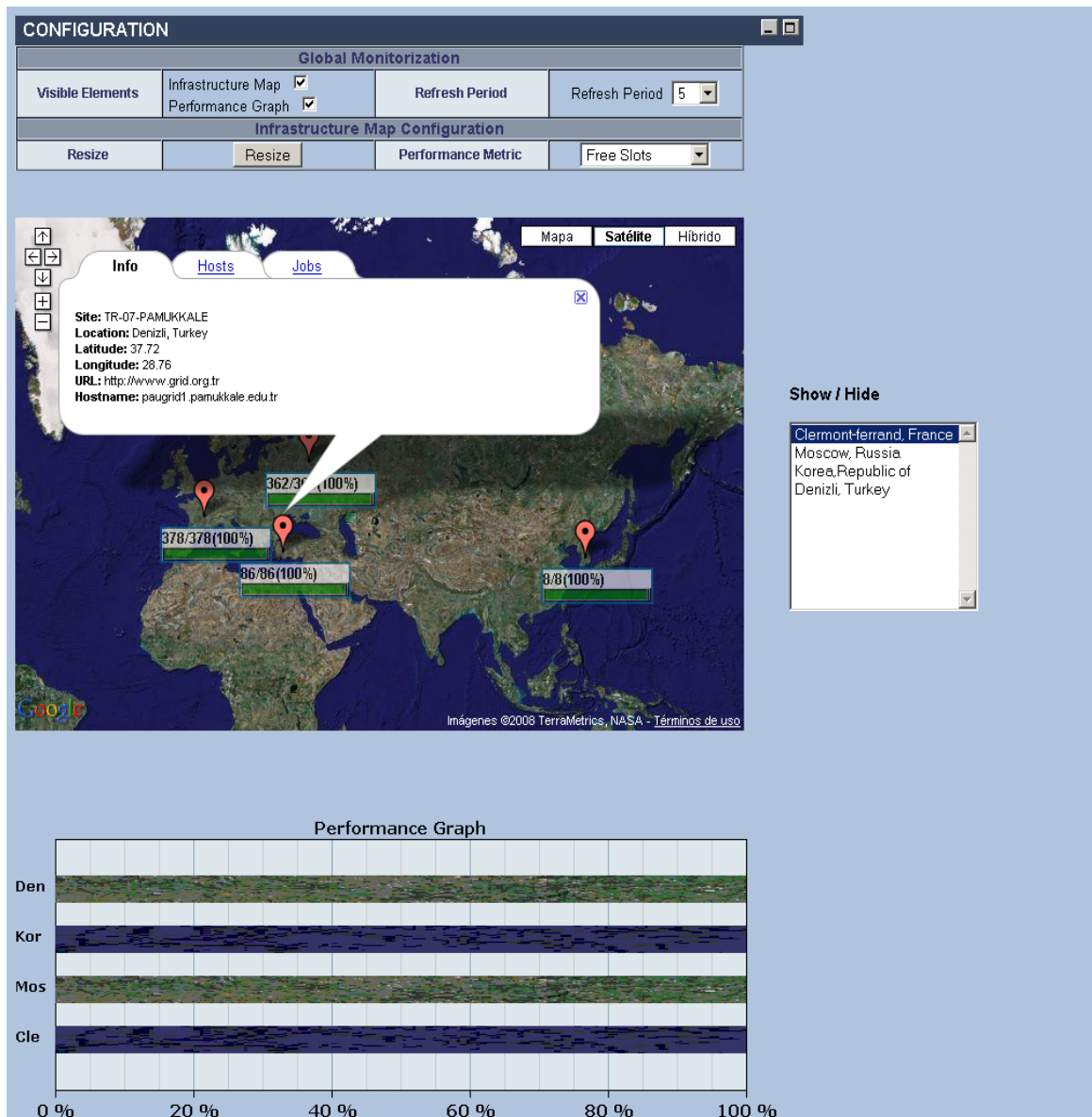
8.2.14. Prototipo 14 (20 de Abril de 2008)

La principal mejora de este prototipo es la modificación del aspecto de la gráfica, dándole la que terminaría siendo su apariencia en la versión final. Además, se eliminan los botones situados a la derecha del mapa, pasando esta área a ser ocupado el exclusiva por el recuadro con la lista de sitios.



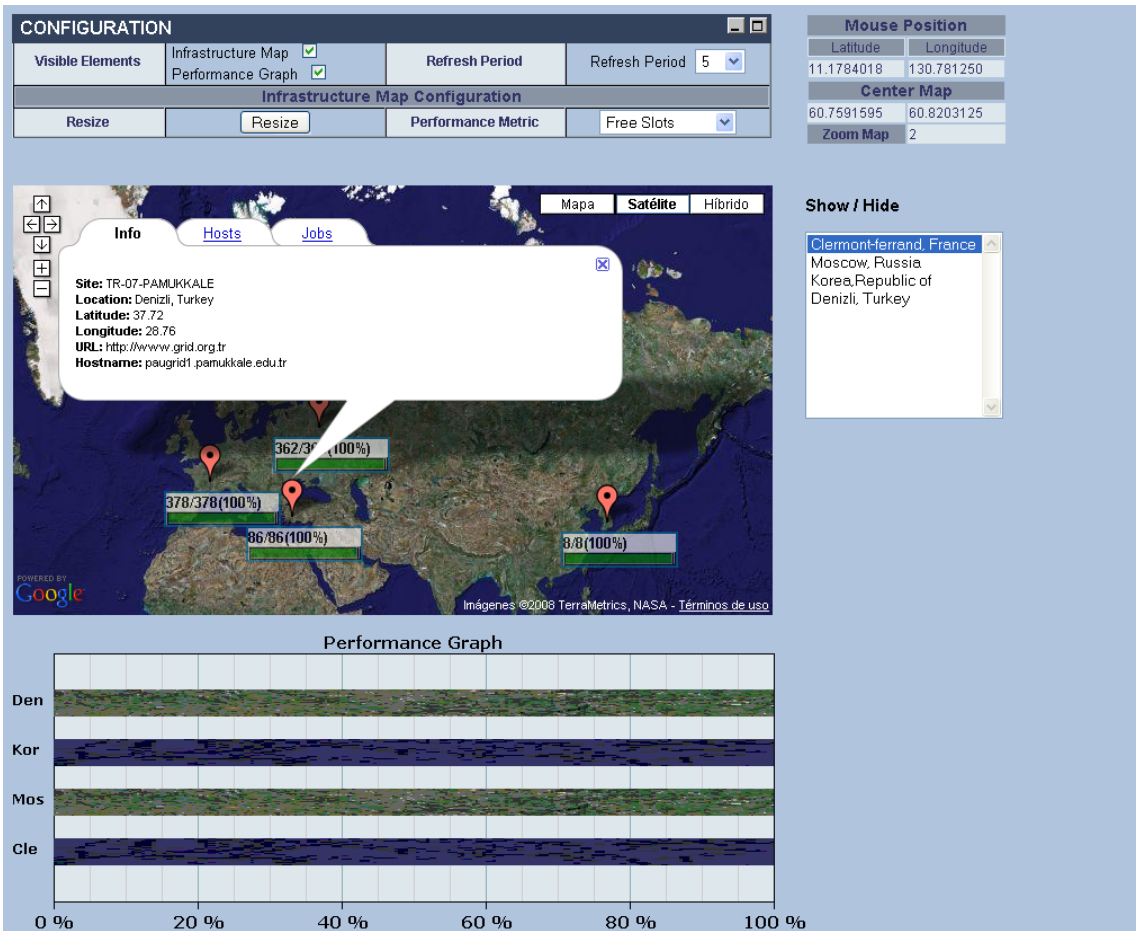
8.2.15. Prototipo 15 (27 de Abril de 2008)

Las modificaciones con respecto al anterior prototipo consisten en la eliminación en el recuadro de configuración del apartado “Sites” (de este modo, hacemos que la ocultación de sitios sólo se pueda realizar a través del recuadro situado a la derecha del mapa) y la modificación del código para arreglar algunos “bugs”, entre los cuales destacaba un fallo que hacía que no se calculasen los porcentajes correctamente.



8.2.16. Prototipo 16 (15 de Mayo de 2008)

Presenta como gran novedad la aparición de un recuadro de información cartográfica en el que se indican las coordenadas del mapa sobre las que se encuentra el puntero del ratón, las coordenadas del centro del mapa y el nivel de zoom de éste. Además, esta versión corrige algunos fallos relacionados con la recolocación de los elementos de la aplicación al ocultarlos y volverlos a mostrar mediante las opciones del apartado “Visible Elements” del recuadro de configuración.

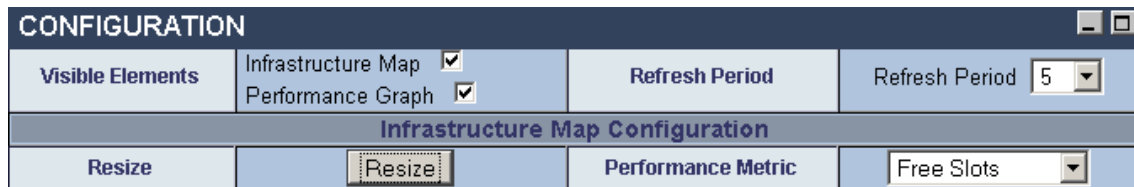


9. DESCRIPCIÓN DE LA INTERFAZ

La presente aplicación web para la geolocalización y monitorización en tiempo real de los recursos integrantes de una red Grid pretende ser una herramienta extremadamente poderosa, pero sin que ello repercuta negativamente en su forma de un manejo excesivamente complicado. De esta forma, hemos conseguido una aplicación cuyo manejo resulta altamente intuitivo, tal y como se verá en las siguientes páginas.

Una vez se tiene en funcionamiento la página web, podemos observar en ella cinco elementos claramente diferenciados: el recuadro de configuración, la ventana de información cartográfica, el mapa de visualización, el recuadro de selección de sitios y la gráfica de monitorización. Las características de cada uno de estos elementos se describirán detalladamente en los siguientes apartados.

9.1. Recuadro de configuración



Supone el modo que tiene el usuario de interactuar con la página web para modificar su funcionamiento. Además de los dos botones situados en la parte superior derecha para la minimización y maximización del propio recuadro, contiene una serie de elementos (descritos brevemente en los siguientes apartados) cada uno de los cuales se encarga de una tarea concreta dentro de la configuración de los elementos de la página.

9.1.1. Visible Elements

Permite indicar cuales de los elementos de información (mapa y gráfica) se muestran en la aplicación. De esta forma, el usuario puede dejar la opción por defecto de que se muestren ambos elementos, hacer que sólo se muestre uno de esos elementos e incluso hacer que no se muestre ninguno de los dos; para hacer cualquiera de estas elecciones, basta con seleccionar o deseleccionar las casillas de “Infrastructure Map” y “Performance Graph”.

9.1.2. Refresh Period

Permite seleccionar cada cuanto tiempo se refresca la aplicación, es decir, cada cuantos segundos se actualiza la información mostrada por el mapa y la gráfica. Permite seleccionar entre 5, 10, 15, 20, 25 y 30 segundos.

9.1.3. Resize

Este apartado consta de un botón que permite que si se ha realizado alguna modificación en el mapa (se ha modificado el nivel de zoom o se ha desplazado el área a mostrar), este vuelva a las condiciones iniciales.

9.1.4. Performance Metric

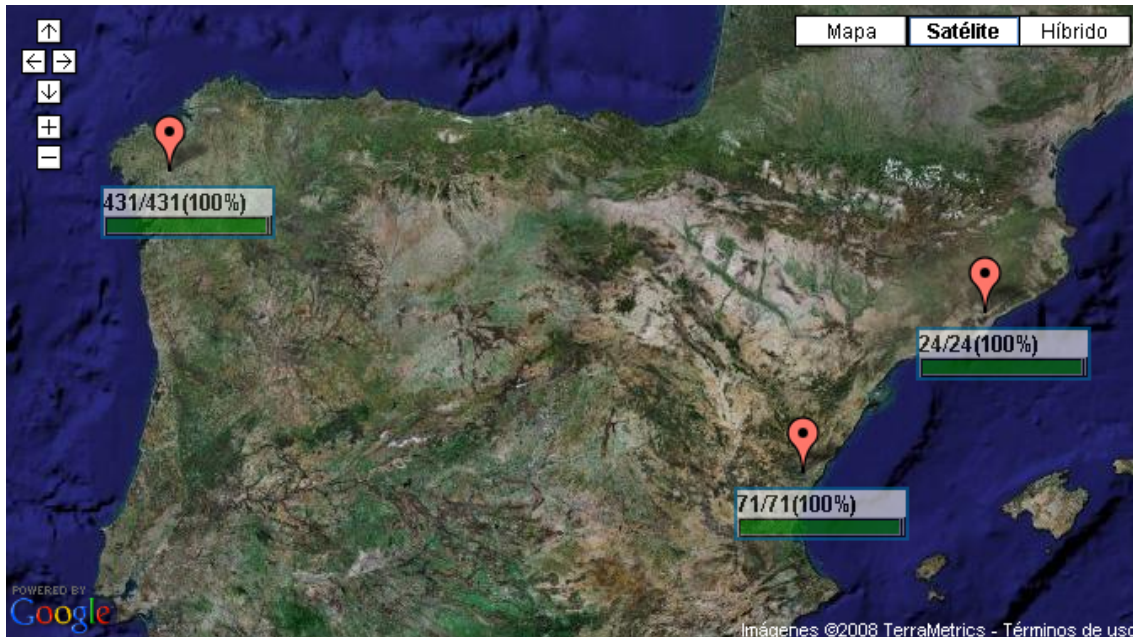
Permite seleccionar el tipo de métrica a mostrar tanto en la gráfica general como en la de los elabels del mapa. Permite seleccionar las métricas “Free Slots” (número de slots libres con respecto al número total de slots), “Used Slots” (número de slots usados con respecto al número total de slots), “Pending Jobs” (número de trabajos pendientes con respecto al número total de trabajos) y “Executing Jobs” (número de trabajos en ejecución con respecto al número total de trabajos).

9.2. Ventana de información cartográfica

Mouse Position	
Latitude	Longitude
42.2773087	4.10888671
Center Map	
41.4015000	-3.2325000
Zoom Map	6

Muestra una serie de elementos de información geográfica, como son las coordenadas sobre las que se encuentra localizado el puntero del ratón, las coordenadas del centro del mapa y el nivel de zoom que se aplica en ese momento al mapa. Es un elemento meramente informativo con el que, por tanto, el usuario no puede interactuar.

9.3. Mapa de visualización



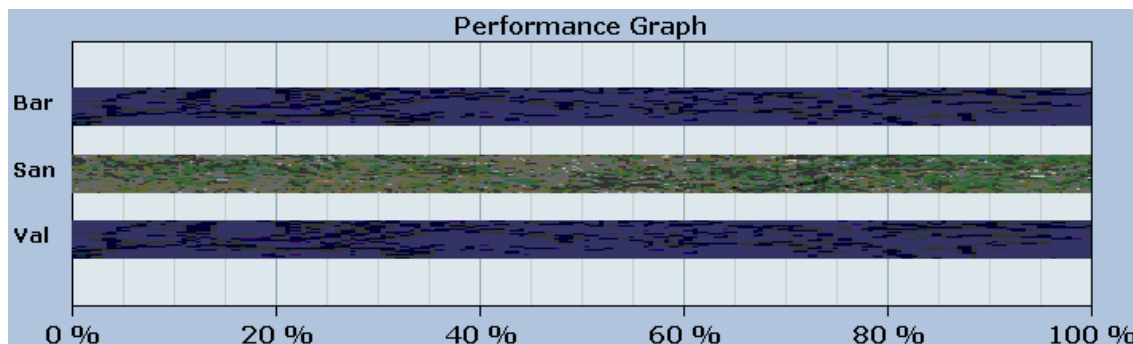
El elemento principal de la aplicación es un mapa obtenido a través de la herramienta Google Maps. Permite las mismas opciones de interacción que cualquier otro mapa de Google Maps: desplazar el área a mostrar (manteniendo pulsado el ratón mientras se arrastra o mediante las flechas situadas en la esquina superior izquierda del mapa); cambiar el nivel de zoom (haciendo doble clic con el botón izquierdo del ratón para aumentar el zoom o con el botón derecho para disminuirlo, o bien pulsando los botones con los símbolos del más y el menos situados en la parte izquierda del mapa) ; y cambiar el tipo de mapa que se muestra (seleccionando entre “Mapa”, “Satélite” e “Híbrido” con los botones de la esquina superior derecha del mapa). Además, sobre cada localización geográfica con participación en la red grid se muestra un marcador que al ser pulsado hace que aparezca una ventana con información de monitorización distribuida en tres pestañas, de tal forma que para pasar de una pestaña a otra basta con hacer clic en el nombre de la pestaña que deseamos abrir; por otro lado, para cerrar una ventana se puede elegir entre pulsar en el botón de cerrado situado en la esquina superior derecha de la ventana, o bien entre hacer clic en cualquier parte del mapa.

9.4. Recuadro de selección de sitios



Permite seleccionar que sitios de la red grid queremos visualizar en el mapa y en la gráfica. Para ocultar un sitio, basta con pulsar sobre el nombre del sitio visible que queremos ocultar; mientras que para mostrar un sitio, se ha de pulsar sobre el nombre del sitio no visible que se desea mostrar. Este recuadro, además, tiene una función informativa, ya que indica los sitios no visibles mediante la aparición de la palabra “hidden” al lado del nombre de los sitios no visibles.

9.5. Gráfica de monitorización



Sirve para mostrar de un modo visual la información acerca de cada uno de los sitios. Así, para cada sitio (representado con las tres primeras letras de su nombre) aparece una barra orientada horizontalmente y cuya longitud depende del valor del porcentaje para dicho sitio (lo que marca dicho porcentaje viene dado por el tipo de métrica seleccionada en el recuadro de configuración). Al igual que la ventana de información cartográfica, es un elemento meramente informativo con el que el usuario no puede interactuar.

10. SIMULACIÓN DEL FUNCIONAMIENTO DE LA APLICACIÓN:

Esta aplicación esta pensada para funcionar integrada dentro de una máquina, llamada “draco”, perteneciente a la VO-DACYA-UCM y gracias a la cual se pueden ejecutar los comandos cuya información se quiere monitorizar, ya que desde dicha máquina se proporciona acceso a GridWay.

Se tuvo acceso a esta máquina prácticamente desde el principio del proyecto, gracias a un nombre de host y de usuario, así como la contraseña de usuario que se nos proporcionó. Sin embargo en un primer momento, este acceso fue usado únicamente para familiarizarnos con la información que obteníamos de la ejecución de los comandos que debíamos monitorizar.

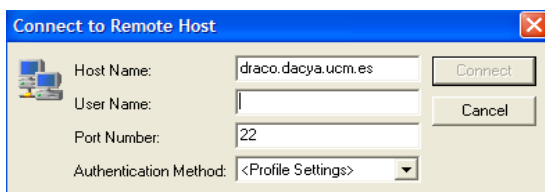


Figura: Conexión a la máquina remota

Los primeros prototipos (del 1 al 10) simulaban el funcionamiento de la aplicación a través de un bucle en el que se generaban números aleatorios del 1 al 100, emulando estos el número de jobs que se estaban ejecutando con respecto a número de jobs que estaban en espera de ejecución.

A partir del prototipo 10, nuestra aplicación leía los datos a monitorizar de un fichero .xml. El problema era que, dado que el fichero xml era estático, sabíamos que la aplicación leía bien los datos y los mostraba correctamente, pero desconocíamos si esta era receptiva a cambios que se produjeran en ese archivo.

Para acercarnos aún más al funcionamiento que esta aplicación debería tener en su versión final instalada en la máquina, se crearon dos demonios, que se lanzaban con 10 segundos de diferencia, y cada uno de ellos modificaba los ficheros xml de entrada de datos. Se comprobó que la salida reflejaba los cambios producidos en los ficheros de entrada.

El paso siguiente fue ya instalar la aplicación en “draco”, para lo cual Javier Fontán, nos creó un directorio en el que debíamos colocar los ficheros y nos proporcionó una url para poder acceder a nuestra web. Además nos preparó una instalación GridWay para poder ejecutar los comandos. Días mas tarde, al ver que usábamos ficheros .cgi, nos preparó también un directorio “cgi-bin” para ejecutarlos. Así que, por una parte estaban los .cgi encargados de ejecutar los comandos, recoger su salida, y transformarla a .xml, y por la otra, nuestra aplicación html con JavaScript que recogía estos xml y mostraba la información que le llegaba.

Hasta entonces en nuestras simulaciones habíamos seguido con los dos demonios, que modificaban los xml. Así que había llegado el momento de dar el último paso para que la instalación estuviera en la versión definitiva. Este consistía en añadir al crontab

instrucciones para ejecutar los comandos y que estos se volcaran a ficheros de texto, y después de esto invocar a los .cgi encargados de leer de estos ficheros y generar los xml. Sin embargo en algo tan simple como la ejecución de los comandos y que estos fueran volcados a ficheros de texto, estuvimos atascados bastante tiempo, ya que los ficheros de texto que obteníamos estaban vacíos. Esto resultaba bastante extraño ya que a través del terminal si que veíamos la ejecución de los mismos. Además, si la información que volcábamos a ficheros, era la de otro comando, por ejemplo pwd, que muestra el directorio actual, esta si que se guardaba en el fichero.

Así, que vía mail, consultamos a Javier este problema y nos comento que se trataba de un problema de entorno.

Cuando se hace login a la máquina, se carga un entorno que con variables que los comandos GridWay necesitan (PATH, GW_LOCATION, GLOBUS_LOCATION...). En cambio, cuando se ejecuta algo desde el cron, este entorno no se carga con lo que programas fallan.

Podíamos ejecutar correctamente el pwd, ya que no necesita ningún tipo de entorno para funcionar. El directorio donde se encuentra está en el PATH por defecto y no hace falta nada más.

Nos recomendó que en lugar de ejecutar directamente nuestro script, nos creáramos un wrapper, que es un shell script que se encarga de cargar el entorno para luego ejecutar el script que necesitáramos.

Creamos este wrapper, que sin embargo, a día de hoy aún no hemos podido probar su funcionamiento, ya que la ejecución de los comandos comenóa dar error como salida, y, finalmente hemos dejado de tener acceso a draco.

Comentamos esto con Ignacio Martín, y nos dijo que usáramos unas trazas con la salida de la ejecución de los comandos, que nos fue guardando un compañero de Javier.

10.1 Simulación a través de trazas

Para la simulación con estas trazas hemos tenido que modificar los .cgi's que generaban los xml, haciendo que hagan un recorrido a través de todos y cada uno de los ficheros salida de la ejecución de los comandos que forman la traza, y dar la salida xml resultante de procesar los mismos con cierto retardo para que la visualización se pueda realizar de forma correcta.

Dado que estos nuevos ficheros que conforman la traza contenían hosts distintos a los que estábamos usando, tuvimos que modificar los ficheros de agrupación y de "información per site" que funcionaban en la anterior versión.

Así que en principio, nuestra aplicación lo que hace es monitorizar la salida de unos comandos que fueron guardados en ficheros de textos, y que son transformados a .xml con un cierto retardo para ser visualizados en la web.

11. INSTALACIÓN Y EJECUCIÓN DE LA APLICACIÓN

11.1. Instalación de la aplicación (únicamente si se ejecuta con las trazas)

Tal como ya se ha comentado anteriormente, la imposibilidad de acceder a la máquina remota “draco”, nos llevó a la utilización de trazas con la salida de la ejecución de los comandos para acceder a los datos, lo que permite la utilización de esta herramienta instalándola en un computador con un sistema operativo Unix

Esta instalación sólo será necesaria si la aplicación se ejecuta a través de las trazas, ya que, en un principio, esta aplicación está pensada para su utilización a través de la red; la forma de instalar esta aplicación se describe continuación:

- 1) Se debe descomprimir el archivo de la aplicación en una carpeta cualquiera.
- 2) A través del terminal de Linux; se debe acceder a la carpeta “cgi-bin” de la aplicación, y después ejecutar la orden “./demonio.cgi&”.

11.2. Ejecución

Si la aplicación no se encuentra instalada en la máquina, para su ejecución se debe acceder a la carpeta “html” y abrir el archivo con extensión “.html” almacenado en dicha carpeta.

En caso contrario, para acceder a la web de la aplicación, bastará con escribir en el navegador la url <http://draco.dacya.ucm.es/users/tutorial01/aplicacion.html> .

12. POSIBLES AMPLIACIONES Y MEJORAS

Habiendo alcanzado los objetivos propuestos para el desarrollo de este proyecto, cabe la posibilidad de pensar en ampliaciones del mismo, que podrían ser realizadas en futuros proyectos.

Proponemos las siguientes:

En lugar de que las agrupaciones de hosts por sitio vengan dadas de manera estática en ficheros, se podría, de modo interactivo, crear a través de la propia aplicación sus propios sites, dándoles un nombre, coordenadas y añadiéndoles los hosts deseados.

Podemos pensar en ampliar también el número de estadísticas mostradas así como el formato de las mismas (tablas, diagramas lineales con la evolución de la carga de trabajos, etc).

Podemos convertir una aplicación que se dedica únicamente a la monitorización de recursos, en una herramienta en la que se puedan gestionar estos recursos. De modo que el usuario podrá iniciar, suspender o eliminar jobs que se están ejecutando, a través de la aplicación.

Podemos crear un entorno web para el administrador, en el que se podrían controlar entre otros aspectos la activación y parada de los demonios o la generación dinámica o estática de los XMLs de agrupación.

13. AGRADECIMIENTOS

Queremos mencionar a las siguientes personas por la ayuda que nos han proporcionado durante el desarrollo de este proyecto:

- Ignacio Martín Llorente
- José Luis Vázquez-Poletti
- Javier Fontán

14. BIBLIOGRAFÍA Y REFERENCIAS

14.1. Libros

- Holzschlag, M. E.; *La biblia de HTML 4*.
- McFedries, P.; *JavaScript*.
- Wall, L.; Christiansen, T.; Orwant, J.; *Programming Perl*.
- Floyd, M.; *Creación de sitios web con XML*.

14.2. Páginas web

- <http://www.gridway.org>
- <http://code.google.com/apis/maps>
- <http://econym.googlepages.com/index.htm>
- <http://www.ecosvirtuales.com/tutoriales/new/TPerl.pdf>
- <http://www.tecnomaps.com>